UNIVERSIDAD AUTÓNOMA DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

TRABAJO FIN DE GRADO

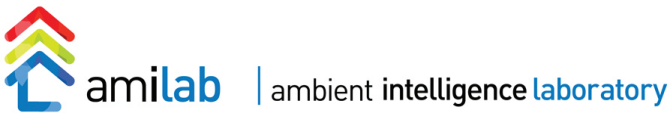
TEI (TWITTER EXPLOITATIONAL INTERACTION): CREACCIÓN DE UNA APLICACIÓN WEB DE VISUALIZACIÓN DE TWITTER

Aída Morcillo Rincón

Julio 2014

Tutor:

Manuel Garcia-Herranz del Olmo



Interesting things happen when
you take what was invisible
and make it visible.

-Jon Kleinberg

Resumen

Existe una gran cantidad de información muy valiosa que fluye a través de las redes sociales. Esta información llega a servir para detectar epidemias, realizar estudios de mercado, análisis de comportamientos, predicción de ataques cibernéticos e incluso sirve como sensor para detectar desastres medioambientales.

El realizar un análisis de este tipo de datos, suele ser una tarea muy compleja, además de requerir un largo tiempo. Por ello, se requieren herramientas que permitan obtener conclusiones de forma más rápida y directa. Con este proyecto se pretende aportar una herramienta de bajo coste que permita al usuario realizar análisis de manera ágil y sencilla, para que, una vez que se ha detectado un evento, se pueda realizar una investigación más exhaustiva.

En esta aplicación se analizan ciertas variables como son los mapas, ya que son una forma de ofrecer una amplia información de forma rápida. A su vez, el grado de los usuarios a lo largo del tiempo, es una forma de utilizar a los usuarios como sensores, de forma que en muchos casos se llegan a predecir eventos futuros.

Otra metodología utilizada para la interpretación de grandes cantidades de datos, es el estudio estadístico, el cual tiene una importante labor de reducción de dimensiones, a la vez que ofrece datos concretos y concisos. Asimismo, la visualización de las mismas permite identificar distintas clases o firmas que caracterizan determinados eventos. Por último, las variables más populares pueden utilizarse como sistema de recomendación, de forma que se le aporte al usuario una visión más amplia de búsqueda.

Palabras clave

Redes sociales, análisis rápido, detección, sensores, clases, bajo coste, reducción de dimensiones, sistema de recomendación, datos concretos y concisos.

Abstract

In the last years online social networks have been acquiring a place of its own in people's daily lives, witnessing an enormous increase in their number of users, the amount of data that is moved through them or the variety of topics they cover. This impressive and steady growth have transformed them in a fundamental tool for people to engage in social activities and keep connected to the world and for scientist to study, understand and predict many different phenomena, from social distress (such as the Arab Spring) to natural catastrophes (such as earthquakes), movie box revenues, behavioral dynamics or disease epidemics, among others.

Twitter, in particular, due to its public policy and its simple structure has gained special attention both among users and scientists as an invaluable tool through which to curate and discover data, opinions and trends.

Nevertheless, performing an analysis of this amount of data is a very complex and time consuming task involving data gathering, processing and analysis. This process, in science, is still done mostly manually and therefore, specially when scouting for new interesting topics and phenomena to deeply analyze later, it could greatly benefit from tools through which easily and rapidly cover several topics. That is, an Interactive Visualization tool for Twitter.

Focusing in the variables that have been used the most in Twitter's scientific literature, this projects presents a web based application whith which to scout, compare, or characterize different Twitter eventsin a simple and easy way. The application handles the data gathering process, allowing the user to query Twitter directly, requesting old events or new ones as needed.

With a strong focus on geolocation and text matching as the most informative variables, the application also compiles and simplifies other information, inspired in Twitter-based scientific analysis, to provide a comprehensive and powerful tool for discovery, along with a Top list of related users/hashtags/urls to enable a continuous exploration.

Among the most interesting variables, the application includes the temporal evolution of number of users, used in the literature both to characterize events as well as to measure the size of information epidemics; of users degree, used to differentiate exogenous from endogenous events as well as to predict the future spread of epidemics; or a glyph based representation of terms, inspired in interactive visualization techniques to detect IP attacks, to visually characterize the nature of an event.

Despite being a functional tool in its own, this project is designed for scalability to serve as an open platform in which to integrate new research, from network analysis to

recommender systems, to gather data (or upload existing data) for new analysis, and to showcase and bring to the public its results.

Keywords

Social Networking, quick analysis, detection, sensors, classes, low cost, dimension reduction, recommendation system, concrete and concise data.

Agradecimientos

Me gustaría comenzar dando las gracias a mi tutor, Manuel, por todo lo que me ha enseñado, el tiempo que me ha dedicado, por ser un gran profesor y mejor persona. A mis compañeros de la carrera, con los que he compartido un largo y duro camino todos estos años, siempre transmitiendo optimismo, y lo que es más importante, siempre dispuestos a ayudar.

A los que ya no están, por enseñarme a luchar y a nunca rendirme. A Alex, a Iris y a Bachi por compartir conmigo momentos increibles y arrimar el hombro cuando se ha necesitado.

A Laura y a Cris, por ser incondicionales. A aquellas personas que han sido mi familia en el Erasmus y han compartido conmigo una de las experiencias más enriquecedoras de mi vida. A Marta, por su apoyo, por ser siempre un motivo para sonreír y por creer en mi.

Finalmente, a toda mi familia, en especial a mis padres y a mi hermana, por todo ese cariño que me han dado y por ayudarme a levantarme siempre que me he caído.

Os doy las gracias infinitamente. Se muy bien que sin vosotros no habría llegado hasta aquí. Este proyecto tiene un trocito de todos vosotros.

Aída Morcillo Rincón
7 de julio de 2014

Índice general

Índice de Figuras

1 Introducción

1.1. Motivación

Las plataformas sociales han manifestado un extraordinario aumento en su popularidad y en su escala durante los últimos años. Una de estas plataformas es Twitter, un servicio de microblogging que permite enviar mensajes a tiempo real. Los mensajes se pueden formar de un máximo de 140 caracteres, y se les conoce como *Tweets*.

Precisamente esta característica se considera uno de los grandes atractivos de Twitter, ya que de esta forma se permite a lo usuarios encontrar Tweets interesantes de una sola ojeada. Esta restricción también favorece a que se sintetice y se mejore la calidad de la información distribuida.

Los Tweets se muestran en la página principal del usuario, y los usuarios pueden *seguir* (suscribirse) a los Tweets de otros usuarios. A estos seguidores se les conoce como *followers*. Los Tweets son públicos por defecto, aunque puede restringirse a determinados seguidores.

Desde que Jack Dorsey creó Twitter en el año 2006, ha experimentado altas tasas de crecimiento. Partiendo de los 5000 Tweets diarios en el año 2007 [34], se ha llegado a 500 millones de Tweets diarios en 2013 [23], y hoy por hoy siguen aumentando las cifras. Mientras que entre los años 2006 y 2009 el volumen de Tweets aumentó un 1400 % [1], y posteriormente fue decrementándose, en 2014 se estima un crecimiento de un 24,4 % [36].

Con estas altas tasas, ha crecido la necesidad de extraer información útil de estas grandes cantidades de datos. Twitter es una herramienta directa y rápida para obtener datos de interés político, económico y social. Además, se ofrecen estos datos públicamente, por lo que cualquier persona puede hacer uso de ellos.

El inconveniente es que resulta muy costoso para los investigadores analizar grandes cantidades de datos. Por ello, se requieren herramientas que permitan realizar análisis de datos de manera menos costosa y más ágil.

En este punto es en el que surge esta aplicación, con objetivos como permitir realizar un *preanálisis* rápido de información que permita categorizar datos o determinar su clase, señalar influencias, determinar agentes endógenos o exógenos, precisar comportamientos, o incluso utilizarse como sensor [20]. De forma que posteriormente pueda realizarse una investigación más en detalle.

Es tan diversa y abundante la información que se propaga por esta plataforma, que en propósitos de investigación se propone utilizarla para detectar de epidemias [2], trastornos sociales, conflictos o como sensor ante catástrofes naturales [24][31]. Además, se emplea para predecir ataques cibernéticos, realizar estudios de mercado [35], económicos, políticos [28], o de otras materias.

Una manera de desentrañar aspectos espaciales, temporales y de actualidad del sistema social a gran escala, es centrarse en los picos de popularidad de los hashtags (*#término*). Son utilizados como anotación social para definir eventos específicos, temas o memes. Se encuentra que la evolución temporal de los hashtags esconde clases discretas de hashtags, y que gracias al seguimiento de su propagación, se puede determinar si un evento es un estímulo exógeno o inesperado, o por el contrario, se asocia a una propagación endógena [25]. Asimismo, otros componentes como URLs, menciones a usuarios o Retweets pueden apotar otra información útil como si se está haciendo eco de ciertos contenidos, si existe una componente social alta o si un asunto es viral.

Los contenidos en linea muestran dinámicas temporales ricas, ofreciendo un diverso contenido que se genera a tiempo real por los usuarios, lo cual incrementa el proceso de enriquecimiento. Los patrones temporales crecen y se desvanecen con el tiempo, de forma que el contenido compite por atención permanente[37]. Las mayores cascadas de datos, analizándolos temporalmente, tienden a ser generadas por usuarios que han sido influyentes en el pasado o que tienen gran número de “followers"[3]. Organizadores, actores destacados nacionales e internacionales, periodistas y otros usuarios activos, influyen en muchos casos en otros usuarios.

La dinámica de las redes sociales facilita la coordinación mediante la activación de cascadas de información que potencialmente llegan a un gran número de personas en corto tiempo. Se han realizado investigaciones que analizan movimientos ciudadanos o revoluciones como *La Primavera Árabe* o *El movimiento 15-M* [26][21], en los que se identifican distintos tipos de usuarios, y con ellos, distintos roles que juegan papeles fundamentales en la distribución de información.

En conclusión, personas de todo el mundo se comunican entre si utilizando dispositivos que registran interacciones, y se tiene acceso a ellas. Para dar sentido a la oleada de datos de medios sociales como Twitter, se requiere emplear ciertas herramientas que hagan posible su estudio. En esta aplicación se emplean herramientas visuales con componentes geográficos, estadísticos, sociales, semánticos para poder sacar conclusiones sobre ellos.

1.2. Objetivos

Debido a los motivos citados anteriormente, la aplicación posee una serie de objetivos que marcan el trayecto de este proyecto:

Navegación geográfica: Permite situar los flujos de información. De el total de Tweets asociados a cada búsqueda, únicamente pueden ser posicionados en el mapa aquellos que sean geolocalizados, es decir, poseen el parámetro de coordenadas activo. Para obtener información sobre ello, se puede observar la estructura de un Tweet en el apartado 3.3.3.

Exploración de Tweets: Lista de los Tweets devueltos en la consulta a Twitter. Cada elemento de la lista se forma por la imagen del usuario, el texto y la fecha del Tweet. Se pueden diferenciar los Tweets geolocalizados de los que no lo están, ya que los geolocalizados poseen un fondo azul y un icono que los identifica.

Estadísticas generales: Las estadísticas generales se muestran de forma visual mediante un hexágono. Cada uno de los vértices refleja distintos datos de los Tweets relacionados con el término buscado. Estos son: Número de hashtags, menciones, Retweets, URLs, geolocalizados y totales.

Evolución temporal: Se muestra mediante una gráfica temporal la actividad que tienen los Tweets del término buscado. Esta gráfica está acotada en el tiempo según las limitaciones que ofrece Twitter a la hora de suministrar Tweets pasados. Para obtener información sobre estos límites, leer el apartado 3.3.1.

Listas destacadas: Hay un "top 5"de elementos del Tweet más utilizados y de los usuarios que más información transmiten sobre el término buscado. Con ello se pretende ofrecer un sistema de recomendación al usuario.

1.3. Estructura de la memoria

2 Estado del arte: Se realiza un estudio sobre las tecnologías utilizadas en la actualidad para visualizar información obtenida de Twitter, así como la justificación del uso de tecnologías de visualización.

3 Diseño: Se plantea el “cómo"de la aplicación. Incluye las herramientas de desarrollo web utilizadas y una descripción detallada del trabajo con mapas y con Twitter.

4 Implementación: Se detalla la arquitectura del sistema y sus respectivos componentes.

5 Casos de uso: Explicación sobre la utilización de la aplicación, así como ciertas conclusiones que se pueden obtener mediante su uso.

6 Conclusiones: Conclusiones que se han obtenido tras finalizar el proyecto.

7 Trabajo futuro: Propuestas para ampliar la utilidad de la herramienta y hacerla más completa. Éstas no se han estudiado ni se han desarrollado todavía.

2 Estado del arte

2.1. Herramientas utilizadas actualmente para visualizar datos de Twitter

Como se introduce en el capítulo 1, recientemente se ha producido una enorme proliferación de datos en bruto que pretenden ser procesados y organizados de forma comprensible y visualmente atractiva para el usuario. Muchas de estas representaciones se integran en servicios web. Esto supone que, en abundantes casos, el lenguaje de programación utilizado en es JavaScript, y que los datos a analizar suelen obtenerse en formato JSON. Estas caracteríticas también se plasman en esta aplicación.

Hay un número indeterminable de formas de visualizar información originada en Twitter. Además, hay mucha información distinta que mostrar y que estudiar. Estos son los dos enfoques que diferencian una aplicación de visualización de otra. ¿ Qué se muestra? ¿ Cómo se muestra?.

¿Qué información se suele mostrar?

Muchas de las herramientas como *Trendsmap* [4], *Hashtagify* [27] o *Twitter Counter*[33], basan su información en tendencias e influencias. Esto es, en hashtags y en el grado de los usuarios. Las tendencias le dan al usuario conocimiento sobre qué temas están difundiéndose más, mientras el grado del usuario ayuda normalmente a observar el grado de difusión de información de un usuario. Este grado permite sacar conclusiones sobre el crecimiento en cuanto al uso que se puede esperar ver de un tema concreto en un futuro próximo.

Además, algunas como *Tweet Archivist* [12], añaden varias combinaciones de datos como las URLs, Retweets, lenguas utilizadas, menciones, imágenes u otros aspectos que puedan aportar información valiosa. Las URLs se ligan a información obtenida fuera de Twitter, mientras que los Retweets indican que hay información puntual que se transmite sin modificarse nuevamente. Indica cuan vírica es una información. En cuanto a las menciones, ofrecen información sobre el grado de componente social que contiene el tema que

se trata. Por supuesto, no pueden faltar el campo básico de texto de los Tweets, que se muestra en distintos formatos y distribuciones.

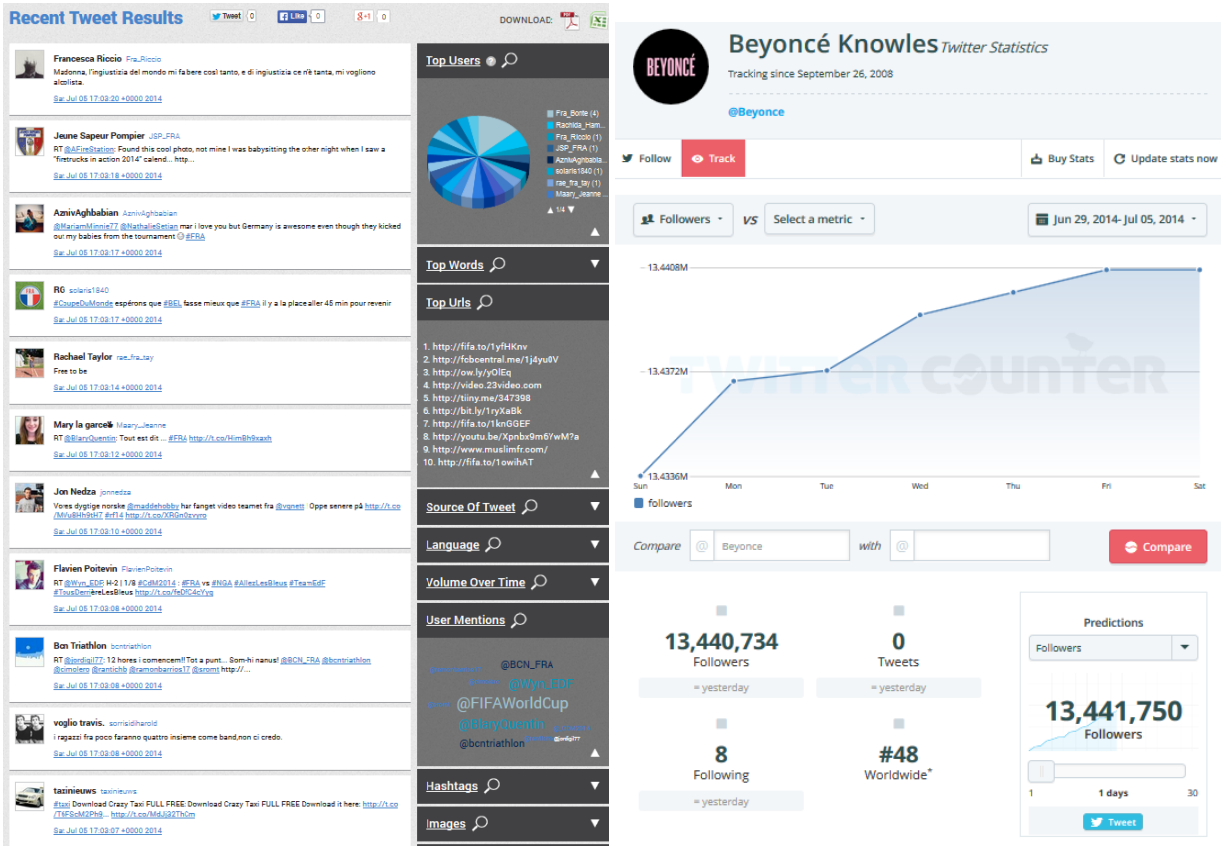
Pueden permitir al usuario, en ciertos casos, introducir aquellos términos de los que quieren obtener información. Este tipo de herramientas ofrecen una información personalizada para el usuario, adaptada a aquello de lo que se desea tener conocimiento concretamente.

¿Cómo se suele mostrar la información?

Una de las formas destacadas de exponer datos es utilizando mapas, como es el caso de *Trendsmap*. Los mapas permiten al usuario identificar los lugares del mundo en los que se están moviendo los distintos flujos de información. Un mapa es una herramienta de gran utilidad siempre que se expongan los datos de manera clara y entendible. El ofrecer un mapa con distintos grados de zoom y ciertas posibilidades de interacción, se consideran aspectos básicos para ofrecer una visualización de calidad al usuario.

Figura 2.1: Ejemplo gráfico de la aplicación Trendsmap [4].

Otra opción destacable, es utilizar conexiones para unir distintos nodos de información, como las que utiliza *Hashtagify* o *MentionApp*[8]. Esto es, en definitiva, utilizar grafos en los que las aristas vinculan los distintos nodos que se muestran. Los nodos pueden ser cualquier tipo de objeto que guarde una relación con otro objeto. En este punto



(a) Ejemplo de visualización de la aplicación Twitter Archivist [12].

(b) Ejemplo de visualización de la herramienta Twitter Counter

Figura 2.2: Ejemplos de visualización de datos en la línea temporal y de forma estadística [33].

se abren muchas posibilidades, podemos hablar de imágenes, hashtags, URLs, usuarios, etc. Es método rápido de buscar elementos relacionados con otros.

Sin duda, los métodos más tradicionales, como son las gráficas lineales, son prioritarias para exponer datos en relación al tiempo. Se puede observar un ejemplo de su uso en *Twitter Stream Graphs* [11]. Probablemente sea una de las formas más claras de visualizar evoluciones de cualquier tipo de dato obtenido. Al igual que las estadísticas clásicas utilizadas por *Twitter Archivist*, presentan información muy precisa, y muchas veces imprescindible para poder sacar conclusiones.

La disposición de los datos en forma de nube suele dividirse en dos opciones, ambas incluidas en *Sentiment viz* [10]. Las nubes de agrupaciones o “clusters", en las que cada agrupación se forma por elementos relacionados, y las nubes de etiquetas o “tags", en las que los distintos elementos suelen tener distintos colores y tamaños que identifican su frecuencia. Las nubes de datos también resultan un método claro para visualizar relaciones entre miembros de agrupaciones.

Figura 2.3: Sentiment viz: Utilizando la nube de etiquetas o “tag cloud"[10].

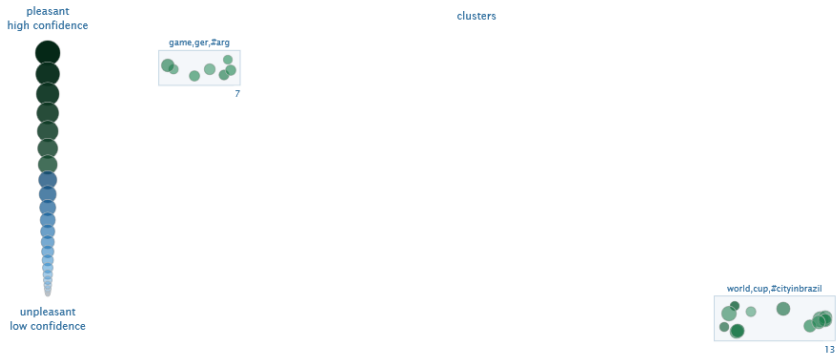


Figura 2.4: Sentiment viz: Utilizando agrupaciones o “clusters"[10].

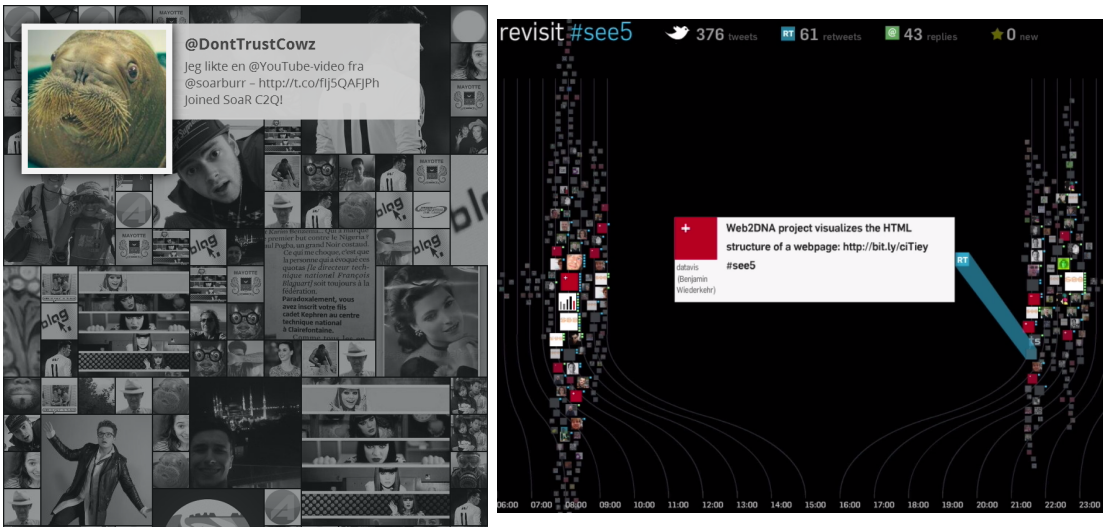
Existen otras metodologías que cada vez se hacen más habituales y atractivas para el usuario, como son la organización de datos mediante imágenes de distintos tamaños según su importancia, como las utilizadas por Tweet Beam [5] o Revisit [9], e incluso algunas aplicaciones, como *Twitt3d* [32], incorporan visualización de datos en 3D (tres dimensiones).

2.2. Análisis de datos

El análisis de datos a nivel temporal permite extraer información representativa sobre las relaciones entre los datos e interpretarla. De esta forma, se pueden estudiar diversos comportamientos que todavía no han sido observados, bien porque todavía no han ocurrido, o simplemente son datos que no se han podido analizar con otros métodos.

Uno de los usos más frecuentes de las series de datos temporales es su análisis para predicción y pronóstico. Mediante el estudio de la evolución de la popularidad de los hashtags, se pueden vincular los eventos a los que los hashtags representan, a ciertas clases discretas.

Los patrones de actividad que muestran formas simétricas respecto al pico o punto más alto de los datos, suelen asociarse con la propagación endógena del sistema. Por el



(a) Ejemplo de visualización de la aplicación Twitter Tweet Beam [5].

(b) Ejemplo de visualización de la herramienta Revisit [9].

Figura 2.5: Organización de datos utilizando imágenes. Además, la segunda imagen orfena los datos en función de tiempo.

contrario, Los patrones que exponen una cola de actividad después del pico, se asocian a eventos inesperados o estímulos exógenos. En el artículo de Didier Sornette [7], se pueden observar ciertas figuras que ratifican estos hechos.

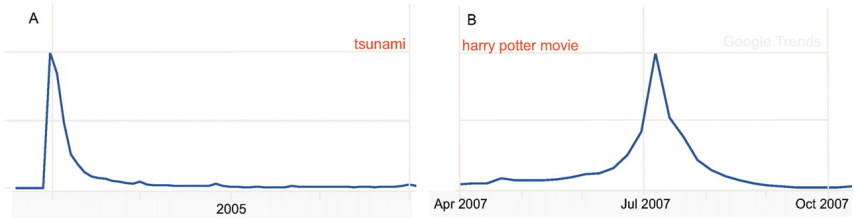


Figura 2.6: (A) El volumen de búsquedas de la palabra“tsunami.[en] las secuelas del catastrófico tsunami en Asia. El pico repentino y relajación relativamente rápida, ilustra la firma típica de una explosión “exógena"de la actividad. (B) El volumen de consultas de búsqueda para “película de Harry Potter". El importante crecimiento que precede al estreno de la película y la relajación simétrica, es característico de una explosión “endógena"de la actividad.

El grado de los usuarios resulta ser una herramienta de detección temprana según el estudio realizado por Manuel García-Herranz [20], en el que se utilizan a amigos como sensores para detectar patrones de estados de ánimo, epidemias, comportamientos de consumidores, y otra serie de eventos. El estudio utiliza la “paradoja de la amistad", la cual consiste en analizar un conjunto de usuarios al azar y algunos de sus seguidores como “grupo sensor". La paradoja de la amistad da a entender que los amigos de un usuario tienen más amigos que él mismo. Los sensores ejercen un papel muy significativo a la hora de propagar la información, ya que reciben la información mucho antes que otros usuarios previamente elegidos.

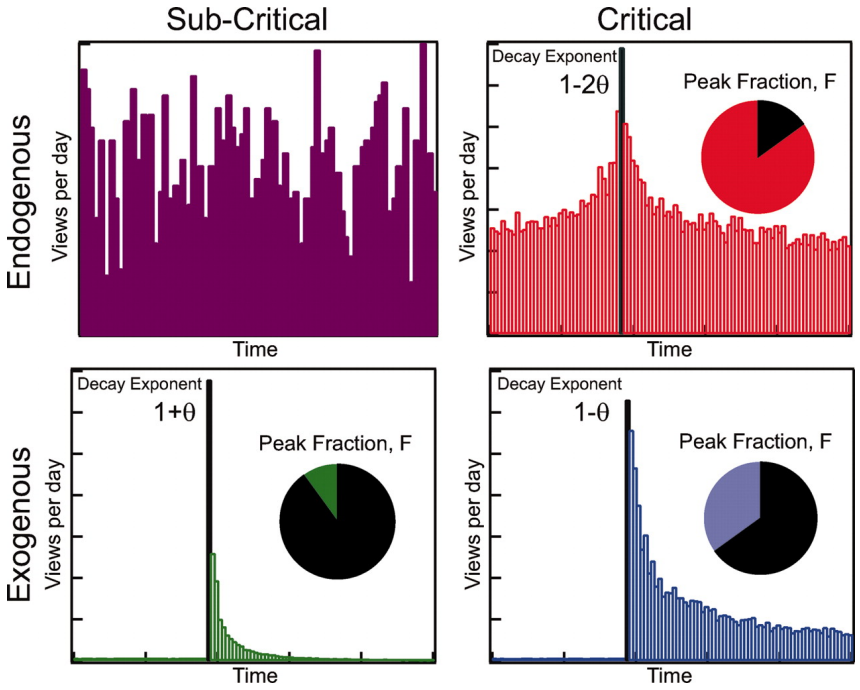


Figura 2.7: Una vista esquemática de las cuatro categorías descritas por los modelos: Endógeno-no crítico (superior izquierda), endógena-crítico (superior derecha), exógena-no crítico (inferior izquierda) y exógena-crítico (inferior derecha). La teoría predice la pendiente de la función de respuesta condicionada en la clase de la perturbación (exógeno/endógeno) y la susceptibilidad de la red (crítico/no crítico).

Este estudio pretende monitorizar la información que se propaga de forma "viral"por la red, que debido a su estructura, los actores altamente centrales, son a la vez más activos y poseen mayor diversidad de información que los demás. Los resultados obtenidos sugieren que el control local resulta más eficiente y eficaz, además de poder aplicar para controlar procesos contagiosos de la red a escala global.

A la hora de utilizar ciertos datos como sensores, un componente crítico es, el análisis de cómo se enruta la información entre los usuarios, los tipo se de usuarios que hay y como se abastecen entre ellos. En el artículo de Lotan [26] relacionado con la revolución democrática árabe, conocida como la “Primavera árabe", se distinguen tres tipos de usuarios:

- Aquellos conectados directamente con algo.
- Aquellos que quieren aprender sobre ese algo, como lectores interesados.
- Medios de comunicación principales que quieren aprender acerca de acontecimientos para ofrecer información y mantener la atención del público.

con esta información pueden surgir muchas preguntas, una de ellas puede ser: ¿Son los periodistas un instrumento estratégico utilizado por organizadores de noticias para trans-

mitir una impresión de personalización?.

Existen ciertos personajes públicos como periodistas, actores influyentes, grandes organizaciones o ciertos usuarios activos, que manejan flujos de información que se extienden a un gran número de personas. Se considera que, en muchos casos, existen fuertes influencias de este tipo de usuarios al resto, que se considera la mayoría de usuarios. Con esto, se puede notar la importancia que pueden llegar a tener las influencias en los datos y en el resto de usuarios. Por ello, se añade a la aplicación, una serie de listas que contienen elementos clave de los Tweets ordenados por su popularidad.

El artículo de Manuel García-Herranz [6], motiva a utilizar herramientas que reduzcan la magnitud de los datos de forma que se puedan analizar datos de forma más asequible. Se realiza un estudio de usabilidad basado en la visualización interactiva y algoritmos de clustering que revelan eventos anómalos de una red a tiempo real. Un 89,37 % de personas independientes al contexto, fueron capaces de identificar ataques al sistema con una precisión del 94,36 % (ver figura 2.8). Por ello, se pretende que, utilizando herramientas sencillas de visualización interactiva, tanto personas dentro del contexto de la investigación, como personas que no lo están, puedan hacer uso de la aplicación y obtener conclusiones de manera intuitiva. Para poder alcanzar este fin, es necesaria una reducción de dimensionalidad, de forma que se hagan visibles los aspectos más relevantes del elemento a analizar.

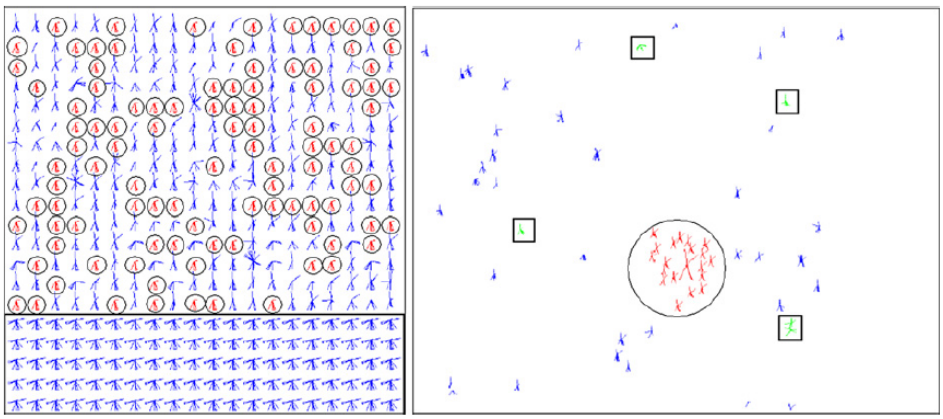


Figura 2.8: Herramienta de visualización interactiva que utiliza glifos y algoritmos de clustering para reducir la dimensión de datos, con el fin de detectar eventos anómalos en una red.

Con el fin de reducir estas dimensiones de la información obtenida en Twitter, se han utilizado visualizaciones estadísticas. Con ellas, se pueden realizar comparaciones más detalladas y concretas entre los distintos parámetros estudiados. Lo que se pretende, es hacer una combinación entre un elemento que ofrezca una visión clara y atractiva, junto

con ciertas medidas concisas que complementen la funcionalidad.

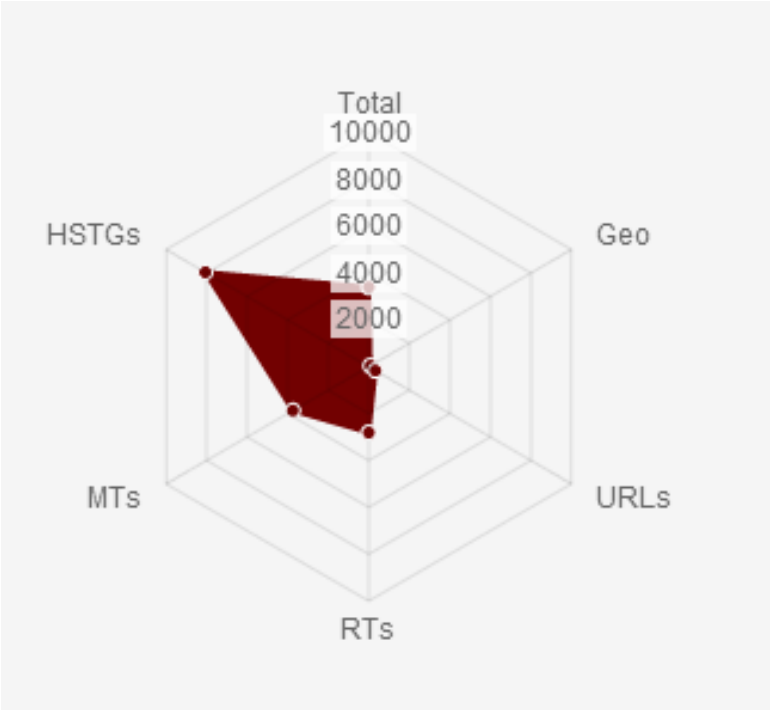


Figura 2.9: Reducción de dimensión de datos estadísticos utilizando un gráfico hexagonal.

3 Diseño

En esta sección se da una explicación detallada sobre el proceso de diseño de la aplicación. Se describe su funcionamiento incluyendo los distintos componentes y decisiones que son fundamentales para alcanzar un diseño robusto.

Este proceso consiste en abordar el "cómo", dando respuesta a las cuestiones que plantean las distintas funcionalidades introducidas en el capítulo 1, de manera que se obtenga un modelo más cercano a la implementación.

El trabajar con mapas, con Tweets, que no se cargue una página web entera cada vez que un usuario realice una acción o la utilización de gráficas para mostrar información, son aspectos del proyecto que exigen el manejo de tecnologías muy diversas.

Se comienza exponiendo los lenguajes de programación utilizados en el desarrollo, enfatizando las razones por las que son elegidos. Se continúa exponiendo el trabajo con mapas, algunas alternativas y aquello que nos puede aportar la visualización geográfica de datos. Finalmente, hay una descripción del módulo más significativo, Twitter, las dos APIs que ofrece, sus limitaciones y qué tipo de información se puede obtener de los Tweets.

3.1. Desarrollo web

La elección del lenguaje de programación es de vital importancia en este proyecto. Existen diversos lenguajes que permiten obtener información de Twitter o gestionar mapas de Google Maps, pero depende de las funcionalidades que se pretendan ofrecer, el lenguaje puede ser una herramienta para alcanzar una meta de manera más asequible y optimizar el rendimiento, o ser todo lo contrario.

En este caso los lenguajes de programación utilizados son:

1. **HTML:** Este estándar sirve de referencia para elaborar páginas web. Al tratarse de una aplicación web, se utiliza este lenguaje para definir un código y una estructura básica del contenido de la misma.

2. **PHP:** La aplicación exige que ciertas operaciones, como puede ser la autenticación del usuario, no sean visibles para el cliente y para el navegador web, lo que provoca

una necesidad de utilizar un lenguaje como PHP, del lado del servidor. Esto conlleva que es el servidor el que ejecuta el código y envía el resultado HTML al navegador. De esta forma, este lenguaje permite una programación confiable y segura. Además cabe destacar lo siguiente:

- Es **software libre**, por lo que se considera una alternativa de fácil acceso.
- Facilita la **escalabilidad**. Se quiere realizar una captura de datos en el servidor para poder guardarlos. De esa manera, se pueden hacer un análisis de datos incluso de años atrás.
- Es posible aplicar técnicas de **programación orientada a objetos**.
- Posee una **amplia documentación**, además de **librerías** para trabajar con Twitter, facilitando el desarrollo de partes del proyecto.
- Es un lenguaje al que se le considera muy **similar a C**, uno de los lenguajes de programación que más se trabajan a lo largo de la carrera. Aunque Php también se ha tratado en la carrera, ha sido en menor profundidad.

3. **JavaScript:** Las razones de más peso por las que se ha tomado la decisión utilizar JavaScript son el hecho de tener acceso directo a todos los elementos de un documento web y que es un lenguaje de scripting del lado del cliente, es decir, que se ejecuta en el navegador o cliente web. Esto implica que es el navegador el que soporta la carga de procesamiento, y de este modo, se libera al servidor de gran parte de la carga de trabajo.

4. **AJAX:** En una aplicación web tradicional, las acciones que realice el usuario (seleccionar una pestaña, pinchar un botón, etc.) originan llamadas al servidor. El servidor se comporta según la petición recibida y devuelve una nueva página web.

   Lo que ocurre en estos casos, es que se desperdicia mucho ancho de banda, ya que gran parte del HTML que devuelve el servidor, no ha sufrido ningún cambio desde la petición y ya estaba presente entonces.

   Otro gran inconveniente, es que en este tipo de aplicaciones se están realizando peticiones continuas al servidor, lo que implica que el usuario tiene que estar esperando, para cada una de las peticiones, que se recargue la página. Ésto no da ninguna buena sensación al usuario.

   La figura 3.1 puede ayudar a apreciar la diferencia entre utilizar un modelo clásico de aplicación web y utilizar el modelo de aplicación web que propone AJAX:

   Las aplicaciones que utilizan AJAX suprimen la recarga persistente de páginas mediante la introducción de un elemento situado entre el servidor y el usuario. Esta capa, a la que se conoce como **"motor de AJAX"**, mejora sustancialmente la

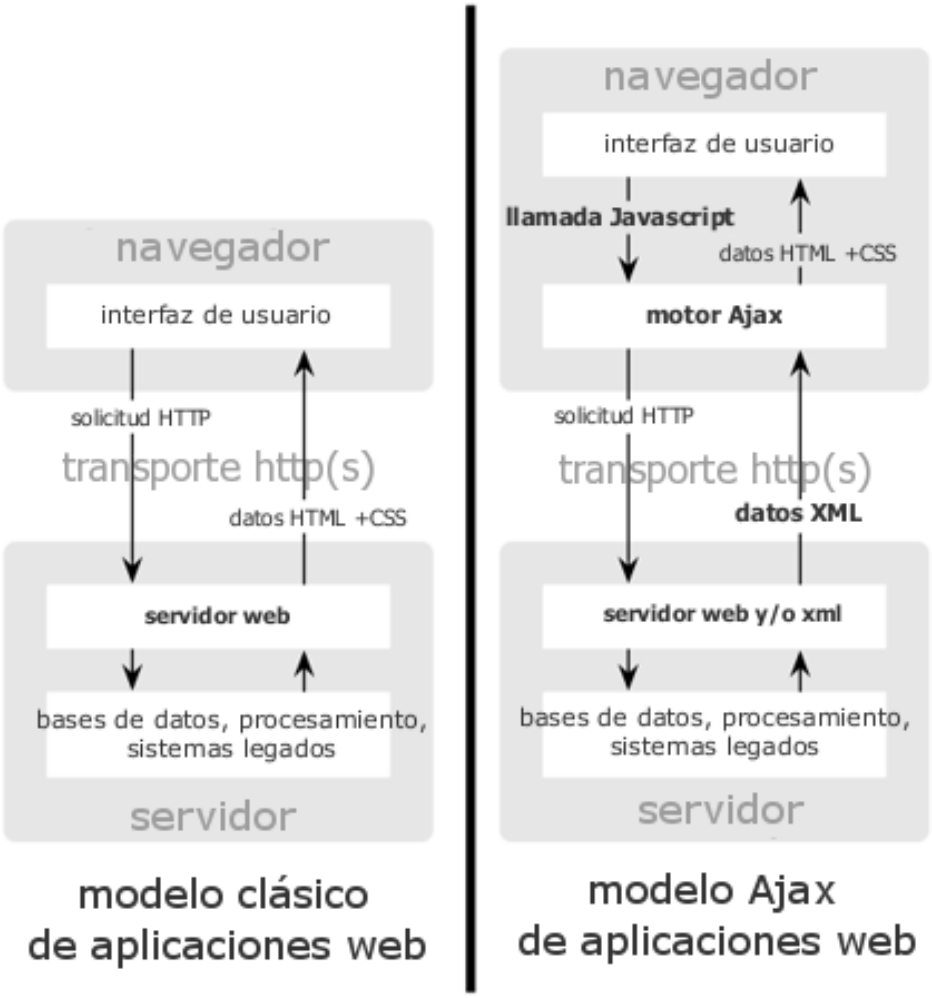


Figura 3.1: Comparación del modelo de aplicación tradicional y el el modelo de aplicación de AJAX

respuesta de la aplicación, puesto que el usuario nunca se encuentra esperando la respuesta del servidor con una ventana del navegador vacía. Este hecho ha sido particularmente decisivo para su empleo en el proyecto.

El motor de AJAX se carga al inicio de sesión, y no es más que un fichero JavaScript que acompaña al HTML. Su trabajo es generar la interfaz que visualiza el usuario y gestionar la comunicación con el servidor, lo cual sucede de manera asíncrona, habitualmente utilizando el objeto *XMLHttpRequest*, y evitando la espera con el navegador en blanco cada vez que se realice un evento, por lo que no se detiene la interacción del usuario con la aplicación.

Otra interesante posibilidad que ofrece, es que hay peticiones tan simples que no requieren que el servidor intervenga. En estas ocasiones la respuesta es inmediata.

5. **JQuery:** Esta librería de JavaScript ayuda a hacer más simple la interacción con documentos HTML, manejar eventos, gestionar animaciones, manipular el árbol DOM

y agregar dinamismo utilizando AJAX. Se utiliza porque ciertas funcionalidades basadas en JavaScript supondrían muchas más líneas de código, por lo que necesita menos tiempo y espacio para lograr resultados.

Resulta muy útil en partes de la aplicación en las que hay que manejar animaciones y en las partes de la aplicación en las que hay que utilizar AJAX. Además de ser compatible con todos los navegadores.

6. **CSS:** Se utiliza Bootstrap, un framework para crear interfaces web con JavaScript y CSS. Su labor es facilitar y agilizar las tareas de dar formato y buen aspecto a una aplicación web. Esto se lleva a cabo gracias a sus componentes predefinidos, la integración de JQuery, su sistema de rejilla para ordenar los elementos de la página y otras ventajas que permiten ahorrar tiempo y trabajo.

3.2. Trabajar con mapas

Una herramienta para visualizar grandes masas de datos puede ser la utilización de mapas. La localización es uno de los datos que aporta Twitter, y es en este punto en el que entra en juego la geolocalización como parte de la aplicación.

Sin duda, es uno de los aspectos destacados de este proyecto. El componente geográfico se precisa para estudiar influencias e intereses políticos, sociales y económicos atendiendo a los lugares en los que se producen o se divulgan.

3.2.1. ¿Qué herramienta elegir?

Los sistemas de información de geográfica o *sistemas SIG* (GIS en inglés) son herramientas para editar, compartir, analizar, almacenar, integrar y exponer información geográficamente referenciada. Estos sistemas posibilitan la creación de consultas interactivas, editar mapas y datos, analizar información espacial y presentar el resultado de todas estas operaciones.

Estos sistemas son utilizados para investigaciones sociológicas, marketing, gestión de recursos, arqueología, ciencia, evaluación de impactos ambientales y en muchos otros ámbitos. Este tipo de mecanismos pueden ayudar a calcular tiempos de respuesta en caso de desastre natural, ubicar nuevos negocios para aprovechar zonas de mercado con menos competencia o incluso encontrar encontrar humedales que requieren protección ante la contaminación.

Un ejemplo de sistema SIG puede ser CartoDB [1], el cual permite crear mapas e integrarlos en páginas web, así como importar datos para poder visualizarlos en mapas siguiendo su evolución temporal. Además, las herramientas de visualización pueden ser

[1] http://cartodb.com/

muy atractivas para el usuario. El principal problema que plantea es que no es una herramienta libre, sino de pago, por lo que se ha descartado.

La mayor parte de los sistemas de información geográfica son herramientas de uso complejo y de pago. Existen otras alternativas a los SIG como son Bing Maps, OpenPaths o Google Earth, pero incluso algunos de estos mapas integrables en lugares web no son libres. El caso de Google Earth fue estudiado, pero no fue escogido porque el proyecto no pretende dar una visualización en tres dimensiones($3D$), además de que resulta ser una herramienta lenta y pesada para los servicios que se pretenden ofrecer.

Versión 3 del API de JavaScript de Google Maps

El API de Google Maps es un servicio gratuito que permite integrar Google Maps en aplicaciones para móviles o en páginas web [13]. Posibilita crear aplicaciones basadas en la ubicación utilizando innovadoras herramientas y servicios de Google. Las características más atractivas son:

- Es una herramienta **estable**, **libre** y **ligera**.
- El API ofrece servicios en **JavaScrip**, lo cual es una buena elección para este proyecto que pretende descargar al servidor de mucha parte del trabajo.
- Otra característica indispensable es que es **fácilmente integrable en aplicaciones web**.
- Se presentan **multitud de servicios** interesantes para una aplicación, además de los considerados básicos o que se aprecian a simple vista en Google Maps. Es un aliciente para tener en cuenta respecto al trabajo futuro y extensibilidad de la aplicación.
- El mapa es **altamente personalizable**. Prácticamente cada elemento que aparece en el mapa puede adaptarse a un estilo requerido. Es posible añadir marcadores de distintos formatos y tonos. Sin olvidar de que se tiene la ocasión de añadir información adicional a cada marcador.
- Un aspecto a tener muy en cuenta es la **amplia documentación** que se ofrece al respecto. Se proponen ejemplos de uso, y la mayor parte de los errores suelen estar documentados en internet. Gracias a ello, se cuenta con un gran respaldo en la fase de desarrollo.

Los servicios ofrecidos por Google Maps que se destacan en el proyecto son los siguientes:

- Introducir y situar marcadores en el mapa, donde cada marcador hace referencia a Tweet geolocalizado distinto.

- Aplicar **distintos valores de zoom** según la amplitud de las zonas que se pretendan explorar.
- **Centrar el mapa** según los intereses del usuario. De manera que cuando se realiza una búsqueda, el centro del que parte el radio de búsqueda es el establecido por el usuario.
- **Obtener las coordenadas** de cualquier lugar del mapa para personalizar la búsqueda de datos en Twitter.
- **Personalizar el mapa** utilizando colores tenues permitiendo que los marcadores destaquen, favoreciendo así la visualización de los datos.

3.3. Trabajar con Twitter

Para poder obtener datos de Twitter se debe realizar una pequeña investigación previa sobre cuales son las opciones que se ofrecen. Existen varias APIs que presentan distintas posibilidades según los objetivos de la aplicación que se va a crear.

3.3.1. APIs de Twitter

Twitter ofrece principalmente dos interfaces de programación de aplicaciones (APIs). Éstas son Streaming API v1.1 y REST API v1.1. Ambas permiten adquirir información de Twitter, pero tienen diferentes limitaciones y se han creado para propósitos distintos [14].

La **API de Streaming**, como su propio nombre indica, suministra una corriente continua de datos que fluye sin interrupción. Esto quiere decir que permite obtener información a tiempo real sin detenciones. No tiene limitaciones para conseguir información de Twitter, pero ¿Es una buena opción para esta aplicación?.

Esta API es para aquellos desarrolladores con necesidades intensivas de datos. Para aquellos que busquen construir un producto de minería de datos o que estén interesados en la investigación analítica.

Se provee a la aplicación de grandes cantidades de datos de forma constante que tienen que ser guardados de alguna forma y en algún lugar para poder ser posteriormente analizados. No obstante, los datos se almacenan desde un momento puntual, y solamente se tiene información desde ese instante, y no desde antes. Esto implica que no es posible analizar datos pasados a no ser que se almacenen durante un tiempo previo, ya sean semanas, meses o años.

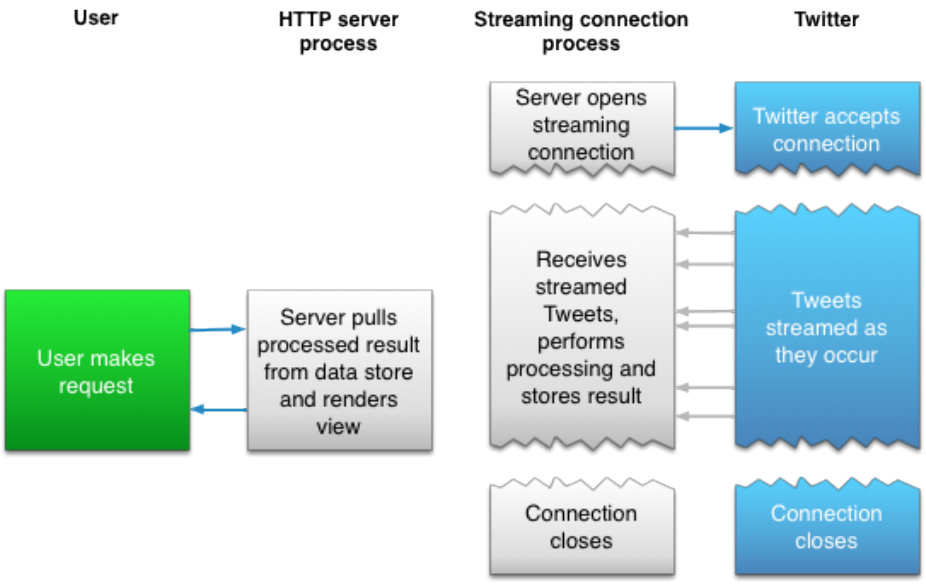


Figura 3.2: Ilustración del funcionamiento de la API de Streaming [29].

Estas características no se acoplan a una aplicación que pretende dar información en presente, pasado y futuro. Por ello, se pasó a analizar otra alternativa.

REST es un estilo de arquitectura para el diseño de aplicaciones de red que se basa en un protocolo de comunicaciones cacheable, cliente-servidor y sin estado. La mayoría de las veces utiliza el protocolo HTTP para realizar llamadas entre máquinas, de forma que es una alternativa a mecanismos más complejos como SOAP. Las solicitudes HTTP son utilizadas para enviar, leer y eliminar datos.

La **API de REST** de Twitter permite acceder a algunas de las primitivas básicas de Twitter, incluyendo líneas cronológicas, actualizaciones de estado y la información del usuario. Se enfoca a aplicaciones que aprovechan los principales objectos de Twitter.

El funcionamiento de la API de REST de una aplicación que acepta solucitudes de un usuario se puede observar en la figura 3.3. Se pueden realizar, una o varias peticiones a la API de Twitter, la cual formatea y devuelve el resultado al usuario en respuesta a la solicitud inicial.

Existe un proceso de manipulación HTTP que hace la consulta a Twitter según las peticiones de los usuarios. Por el contrario, en la figura 3.2 se percibe como la conexión en Streaming se ejecuta en un proceso adicional separado del proceso que maneja las solicitudes HTTP. Éste recibe el flujo de información de Twitter, y es a ese módulo al que se le hacen las peticiones.

Con esto se percibe que la API de REST es más ligera y menos compleja. No es necesario el almacenamiento de grandes cantidades de datos, y a su vez permite al usua-

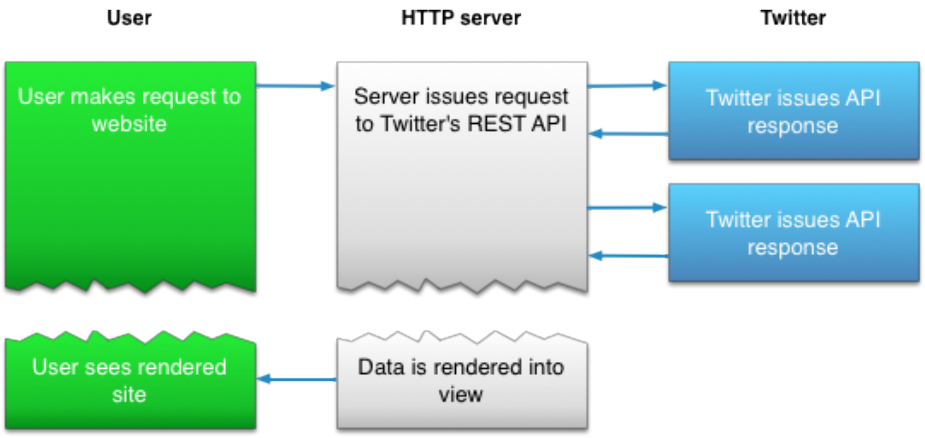


Figura 3.3: Ilustración del funcionamiento de la API de REST [29].

rio realizar peticiones concretas en tiempo real y pasado interactuando directamente con Twitter.

Las consultas a Twitter de las que se ha hablado a lo largo de la sección tienen una estructura fija formada por distintos parámetros que hay que tener en cuenta para poder enviarlos a Twitter y que éste entienda la información que se solicita [19]. Los parámetros son lo siguientes:

q: (Requerido) Búsqueda de 500 caracteres como máximo, incluyendo los operadores. Adicionalmente, las consultas están limitadas por complejidad. *Valor de ejemplo: fiebre.*

geocode: (Opcional) Devuelve los Tweets de los usuarios ubicados dentro de un radio determinado dada una latitud y una longitud. El valor del parámetro se especifica en el orden "latitud, longitud, radio", en donde las unidades de radio deben estar especificadas “mi"(millas) o “km"(kilómetros). Hay un máximo de 1.000 "sub-regiones"diferentes dentro de un radio establecido.
Valor de ejemplo: 40.428656,-3.703194,700km.

lang: (Opcional) Restricción de Tweets a una lengua dada. Para ello se utiliza el código ISO 639-1. *Valor de ejemplo: eu.*

locale: (Opcional) Especificación del idioma de la consulta que se va a enviar.

result_type: (Opcional) Tipo de resultados que se desea recibir. El valor por defecto es "mixed". Entre los valores válidos se encuentran:

- mixed: Incluye resultados populares y a tiempo real.
- recent: Incluye los resultados más recientes.

- popular: Incluye los resultados más populares.

Valor de ejemplo: mixed, recent, popular.

count: (Opcional) Número de Tweets por página. Está limitado a 100. *Valor de ejemplo: 80.*

until: (Opcional) Devuelve los Tweets generados antes de la fecha determinada. Fecha debe tener el formato *AAAA-MM-DD*. Se debe tener en cuenta que el índice de búsqueda puede no llegar tan atrás en el tiempo como indica la fecha que se especifica aquí. Esto se debe a limitaciones temporales de la API, en las que se llegan a conseguir resultados de entre 6 y 15 días atrás. *Valor de ejemplo: 2014-06-01.*

since_id: (Opcional) Devuelve resultados con un ID mayor (más reciente que) que el ID especificado. Hay límites a la cantidad de Tweets que se puede acceder a través de la API. Si el límite de Tweets se ha producido desde el since_id, el since_id será obligatoriamente a la identificación más antigua disponible. *Valor de ejemplo: 12345.*

max_id: (Opcional) Devuelve los resultados que tengan un ID menor o igual al especificado (más antiguos o de la misma fecha). *Valor de ejemplo: 54321.*

include_entities: (Opcional) El nodo entities no se incluirá cuando este valor esté a *falso. Valor de ejemplo: false.*

callback: (Opcional) Si se proporciona, la respuesta se obtiene en formato JSONP. La utilidad de este parámetro ha disminuido por la exigencia de autenticación para las peticiones en esta versión de la API. *Valor de ejemplo: processTweets.*

En esta aplicación no se utilizan todos los campos descritos. Se utilizan *q*, *geocode*, *since_id* y *max_id*.

Aunque esta API sea la que más se adapta a este proyecto, también se han podido examinar una serie de limitaciones. Existe un máximo de consultas por usuario y por aplicación, además de que cada consulta devuelve un máximo de Tweets. Estas limitaciones se tratarán en profundidad en la siguiente sección.

3.3.2. Limitaciones y OAuth

Limitaciones de la API

La API de REST plantea una serie de limitaciones. Para evitar la saturación del servidor, Twitter impone un límite a la hora de realizar solicitudes (queries) de forma continúa.

Según la página oficial de Twitter [16], hay una **limitación de tasa** o de índice que se aplica por usuario y por aplicación. Cuando se excede este límite, el objeto JSON devuelto por Twitter contiene un código de error. Si se continúa haciendo peticiones de forma indiscriminada, ya sea el usuario o la aplicación, será marcado (blacklisted) pudiéndose denegar el acceso a la API de forma indefinida.

Estos límites se dividen en intervalos de **15 minutos de tiempo**. Al contrario que en versiones anteriores, **se requiere la autenticación** de los usuarios para realizar consultas. Ya no existe el concepto de consultas no autenticadas y límites de frecuencia.

En concreto, según los datos obtenidos en la documentación de Twitter[19], la búsqueda está limitada a:

- **180 solicitudes** por usuario en rangos de tiempo de **15 minutos**.
- **450 solicitudes** por aplicación en rangos de tiempo de **15 minutos**.

Como se ha explicado, otros métodos como pueden ser obtener la lista de favoritos de un usuario, obtener la lista de amigos de un usuario, obtener el identificador de un lugar y muchos otros, tienen limitadores distintos al método de búsqueda. Se puede obtener una información más concreta sobre cada uno de ellos en [15].

OAuth

La versión actual de REST API **requiere una autenticación** previa a la consulta a Twitter. Como se acaba de describir en la sección 3.3.2, ya se utilice el identificador de usuario o se utilice el identificador de aplicación, hay ciertos límites al solicitar datos a Twitter. Entre otras cosas, motivado por la necesidad de asociar estos límites a un usuario concreto, Twitter exige que todas las consultas vayan autenticadas. El protocolo utilizado por Twitter para posibilitar las autenticaciones es el protocolo **OAuth**.

Como cada usuario y cada aplicación tiene asociado un identificador de acceso (*access token*), se realiza la restricción por identificadores. Únicamente cuando una aplicación utiliza el método de autenticación con su respectivo identificados, los límites se determinan a nivel global de la aplicación. Este límite es totalmente independiente al límite por usuario.

Esta API ofrece diversos métodos de autenticación utilizando el protocolo OAuth en base a los objetivos de la aplicación como:

- Se quiere añadir un botón de “Registrarse con Twitter".
- **Se quiere leer o enviar datos de Twitter en nombre de los visitantes a un sitio web.**

- Se desea acceder a la API desde la cuenta de un único usuario.
- Se necesita utilizar nombres de usuario/contraseñas y han sido aprobados por XAUTH.
- Se dispone de un ordenador de escritorio o móvil que no puede acceder a un navegador.
- Ofrecer una API donde los clientes envían datos en nombre de los usuarios de Twitter.
- Se desea emitir solicitudes autenticadas en nombre de la propia aplicación.
- Se dispone de una integración basada en iOS5 y se necesitan tokens de acceso para las integraciones del lado del servidor.

Para liberar a la aplicación de algunas de las limitaciones, se ha decidido desarrollar una **autenticación basada en los identificadores de acceso de los usuarios**, es decir, se desea acceder a datos de Twitter en nombre de los visitantes. Esta decisión tiene como fin que las limitaciones recaigan sobre los usuarios según su uso, y no sobre la aplicación entera. De esta forma, aunque ciertos usuarios rebasen la restricciones impuestas, la aplicación esta disponible para otros usuarios que no han rebasado las restricciones de Twitter.

El término para referirse a este protocolo de autenticación es **3-legged authorization** frente a “Sign in with Twitter". Tiene como característica adicional que cada vez que el usuario desee utilizar la aplicación, se le mostrará un mensaje en el que debe de autorizar el acceso. Esto ocurre incluso cuando el usuario ha utilizado previamente la aplicación, excepto si en anteriores ocasiones ha decidido que el sistema le recuerde. (Ver figura 3.4.)

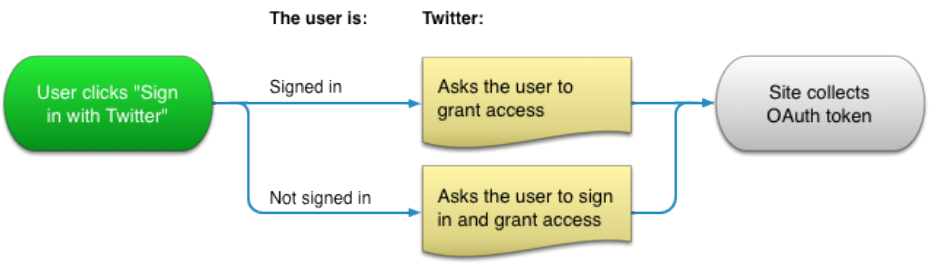


Figura 3.4: Posibles estados en el registro de un usuario utilizando el protocolo "3-legged"[29].

Una vez que se sabe el protocolo de autenticación que se va a utilizar, **¿Cómo funciona el protocolo 3-legged?**[18].

- **Paso 1: Obtener el identificador (token) de solicitud.**
  Para comenzar el proceso, se debe obtener el identificador de solicitud enviando un **mensaje firmado a POST oauth / request_token**. El parámetro único en esta solicitud es *oauth_ callback*, la cual es la URL a la que se desea que el usuario sea redirigido en cuando el paso siguiente. El resto de parámetros son agregados por el proceso de firma OAuth. Se puede observar una explicación gráfica en la figura 3.5.

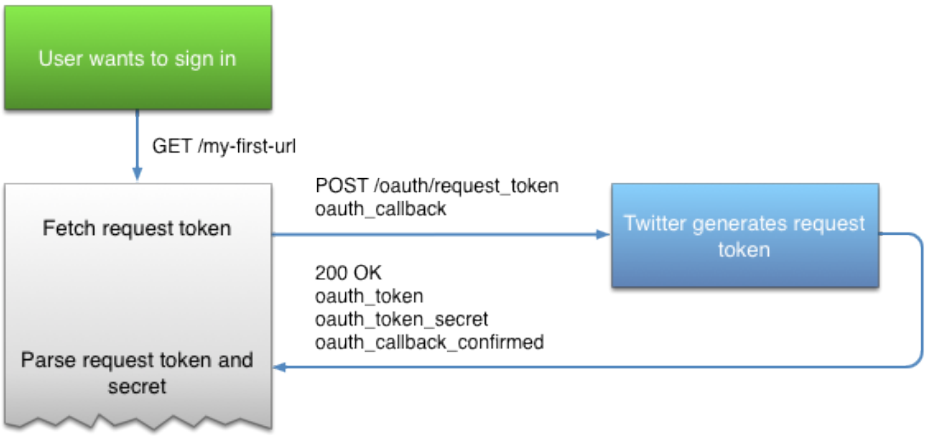


Figura 3.5: Paso 1. Obtener el identificador de solicitud [29].

A continuación, la aplicación examinar el estado de la respuesta HTTP. Cualquier valor que sea distinto a 200 implica que ha habido un error. La respuesta contiene los datos: *oauth_ token*, *oauth_ token_ secret* y *oauth_ callback* confirmados. Finalmente, la aplicación debe comprobar que el parámetro *oauth_ callback_ confirmed* tiene el valor true, y guardar los otros valores para pasos siguientes.

- **Paso 2: Redireccionar al usuario.**
  En la figura 3.6 se aprecia como en este paso lo que se pretende es redirigir al usuario a Twitter para seguir el camino apropiado. Esto se consigue invocando el método GET oauth/authenticate (método para conseguir autenticarse), pasando por parámetro el identificador de autenticación (*oauth_ token*) obtenido en el paso anterior.

  Como se describe en la figura 3.7, según el estado del usuario, el protocolo puede comportarse de tres maneras distintas:

  - Registrado y aprobado: El usuario ha iniciado sesión en el twitter.com y ya ha aceptado la aplicación. Se termina la autenticación correctamente y se redirige al usuario a la URL de devolución de llamada con una solicitud de token de OAuth como válida. La redirección a twitter.com no es visible para el usuario.

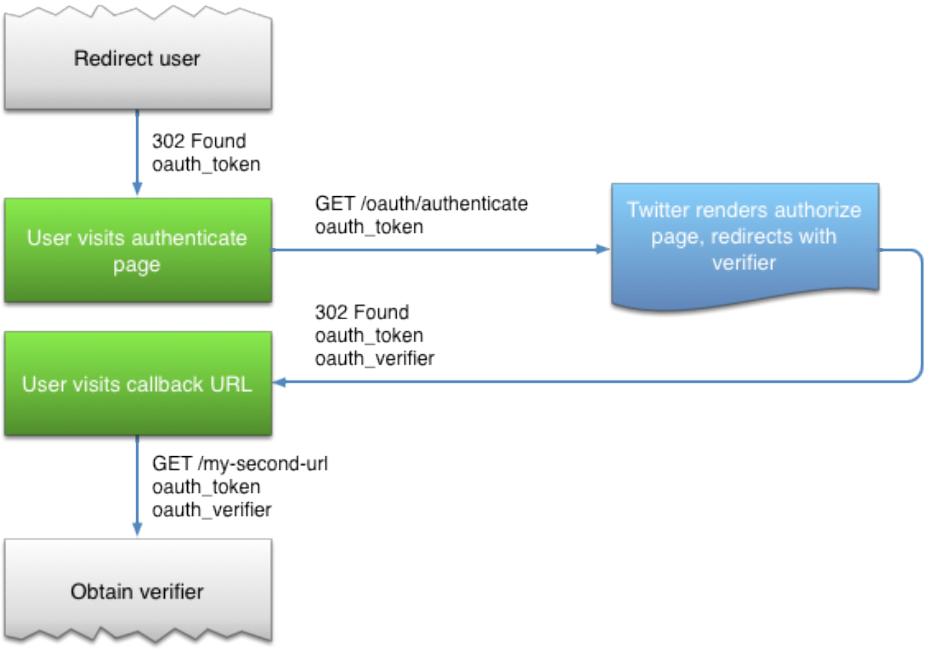


Figura 3.6: Paso 2. Redirección del usuario [29].

- Registrado, pero no aprobado: El usuario ha iniciado sesión en twitter.com, pero no ha aprobado el acceso mediante la aplicación. En cuanto el usuario apruebe el acceso, el usuario es devuelto a la URL de devolución de llamada junto con una solicitud de token de OAuth como válida.
- No registrado: Si el usuario no está registrado en twitter.com, se le pedirá que introduzca sus datos y dará acceso a la aplicación. Una vez registrado, el usuario es devuelto a la URL de devolución de llamada con una solicitud de token de OAuth como válida.

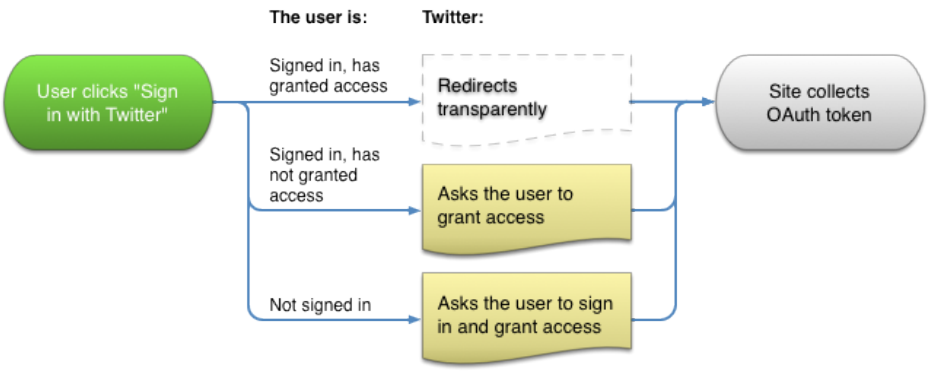


Figura 3.7: Paso 2. Diferentes estados de un usuario [29].

- **Paso 3: Transformar el identificador de solicitud en identificador de ac-**

ceso.

Tras una autenticación correcta, la aplicación recibe una solicitud que contiene los parámetros *oauth_token* y *oauth_verifier*. Se debe verificar que estos datos coinciden con los datos guardados en el primer paso. Finalmente, esto lo consigue la aplicación haciendo una solicitud POST oauth/access_token de punto y final, la cual contiene el verificador obtenido en el paso 2 (*oauth_verifier*).

Como se aprecia en la figura 3.8, una respuesta correcta contiene los parámetros *oauth_token*, *oauth_token_secret* correctamente. El identificador, y el identificador secreto deben ser guardados para posibles solicitudes de autenticación a la API de Twitter.

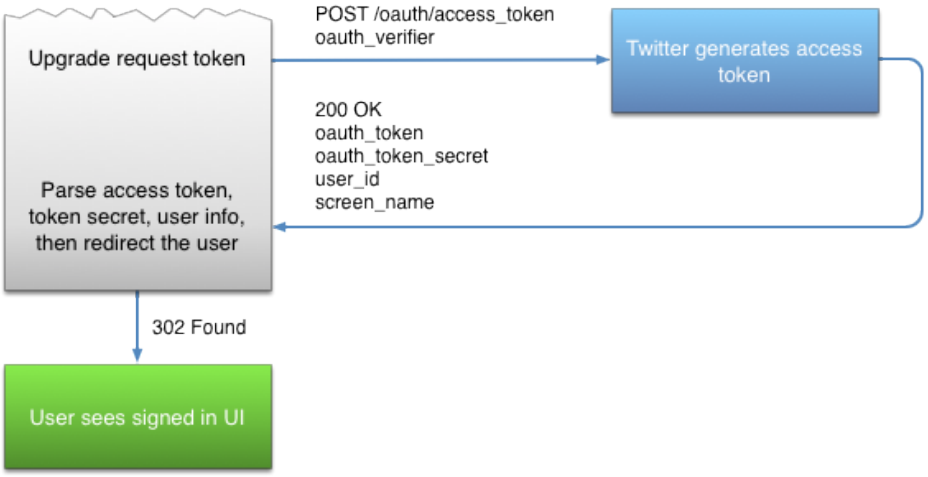


Figura 3.8: Paso 3. Transformación del identificador de solicitud en identificador de acceso [29].

3.3.3. ¿Qué devuelve el API?

Cuando el usuario realiza una solicitud de datos a Twitter, éste envia una respuesta en formato **JSON**. El resultado es un único objeto que se divide en dos campos principales: search_metadata y statuses.

search_metadata: Este campo es un objeto que tiene información sobre la solicitud que se ha realizado a Twitter. Se divide en la siguiente información:

- **completed_in:** Tiempo que tarda en realizarse la consulta a Twitter.
- **count:** Número de resultados que se han solicitado en la consulta.

- **max_id:** Máximo identificador de que se ha obtenido, es decir, el identificador del Tweet más antiguo de la consulta. Sirve de referencia a **next_results** si se pretende hacer una consulta sobre resultados más antiguos. La búsqueda comenzaría desde ese elemento hacia atrás en el tiempo, de manera que Twitter no devuelva de nuevo elementos ya que se han obtenido en solicitudes más recientes.
- **max_id_str:** Representación en forma de String del parámetro max_id.
- **next_results:** Solicitud preparada para obtener los datos que permita devolver Twitter hacia atrás en el tiempo dentro de sus limitaciones. Se puede obtener información de entre 6 y 15 días de antigüedad.
- **query:** El término o términos de los que se ha realizado la búsqueda.
- **refresh_url:** Solicitud preparada para obtener los datos que permita más recientes de Twitter sobre la "query"buscada.
- **since_id:** Identificador del Tweet del que parte **refresh_url** para obtener resultados recientes. Se parte desde el este número de identificador para que Twitter devuelae resultados de ahí en adelante en el tiempo, de forma que no se devuelvan resultados ya obtenidos.
- **since_id_str:** Representación en forma de String del parámetro since_id.

El objeto search_metadata tiene la siguiente forma:

```
completed_in: 0.315
count: 100
max_id: 482293012278820860
max_id_str: "482293012278820864"
next_results: "?max_id=481706140016050175&q=gripe
&geocode=40.428656%2C-3.703194%2C700km&count=100
&include_entities=1"
query: "gripe"
refresh_url: "?since_id=482293012278820864&q=gripe
&geocode=40.428656%2C-3.703194%2C700km&include_entities=1"
since_id: 0
since_id_str: "0"
```

statuses: Es un array con tantos objetos como número de resultados se hayan obtenido. Equivale al mismo número que tiene el **campo search_metadata[count]**. Estos objetos son Tweets. Poseen una estructura muy amplia, por lo que se van a describir aquellos parámetros utilizados en la aplicación o que se consideren de más utilidad. Para una información más exhaustiva, consultar el anexo A o la documentación oficial de Twitter [17].

- **coordinates:** (*Coordinates; Nullable*)Representa la ubicación geográfica del Tweet según lo informado por el usuario o la aplicación cliente. La matriz de coordenadas tiene un formato en el que se sitúa primero la longitud, y a continuación la latitud. *Ejemplo:*

```
"coordinates":
{
    "coordinates":
    [
        -3.6402891,
        40.39208401
    ],
    "type":"Point"
}
```

- **created_at:** (*String*) Momento en el que el Tweet ha sido creado. La fecha y hora se rige por el estándar UTC[2]. *Ejemplo:*

```
created_at: "Thu Jun 26 21:08:01 +0000 2014"
```

- **entities:** (*Entities*) Las entidades que han sido analizadas de forma separada al texto del Tweet. Incluye los hastags utilizados, las urls externas a Twitter y las menciones de usuarios. *Ejemplo:*

```
"entities":
{
    "hashtags":[],
    "urls":[],
    "user_mentions":[]
}
```

- **favorite_count:** (*Integer; Nullable*) Indica cuantas veces aproximadamente ha sido un Tweet marcado como favorito por los usuarios. *Ejemplo:*

```
"favorite_count":632
```

- **favorited:** (*Boolean; Nullable, Perspectival*) Indica si este Tweet ha sido marcado como favorito por el usuario que se autentica. *Ejemplo:*

```
"favorited":true
```

- **geo:** (*Object; Deprecated,Nullable*) Objeto que contiene las coordenadas.
- **id:** (*Int64*) Es la representación entera del identificador único para este Tweet. Este número es mayor que 53 bits. Algunos lenguajes de programación puede tener dificultades para interpretarlo. El uso de un entero con signo de 64 bits

[2]UTC (Coordinated Time Universal): Antiguamente fue llamada "la hora en el meridiano de Greenwich"("GMT") o el "tiempo Zulu"("Z"). Es la hora local en el Meridiano Primario (aquél cuya longitud es 0 grados) dada en horas y minutos en formato 24 horas.

para almacenar este identificador es una forma segura. se utiliza el parámetro *id_str* para buscar este identificador. *Ejemplo:*

```
"id":482269071204184060
```

- **id_str:** (*String*)La representación en String del identificador único del Tweet. *Ejemplo:*

```
id_str: "48226907120418406 4"
```

- **lang:** (*String; Nullable*)Cuando está presente, indica un identificador de idioma BCP 47[3] correspondiente al texto Tweet, o ünd"si el lenguaje no puede ser detectado. *Ejemplo:*

```
"lang": "en"
```

- **place:** (*Places; Nullable*) Cuando está presente, indica que el tweet se asocia a un lugar, pero no necesariamente se origina en ese lugar. *Ejemplo:*

```
"place":
{
    "attributes":{},
     "bounding_box":
    {
        "coordinates":
        [[
                [-77.119759,38.791645],
                [-76.909393,38.791645],
                [-76.909393,38.995548],
                [-77.119759,38.995548]
        ]],
        "type":"Polygon"
    },
     "country":"United States",
     "country_code":"US",
     "full_name":"Washington, DC",
     "id":"01fbe706f872cb32",
     "name":"Washington",
     "place_type":"city",
     "url": "http://api.twitter.com/1/geo/id/01fbe706f872cb32.jso
}
```

- **retweet_count:** (*Int*) Número de veces que el Tweet a sido retwiteado.*Ejemplo:*

```
"retweet_count":322
```

- **retweeted:** (*Boolean; Perspectival*) Indica si este Tweet ha retwiteado por el usuario que se autentica. *Ejemplo:*

[3] http://tools.ietf.org/html/bcp47

```
"retweeted":false
```

- **retweeted_status:** (*Tweet*) Los usuarios pueden ampliar la difusión de los Tweets escritos por otros usuarios mediante un retwit (retweet). Los retweets se pueden distinguir de los tweets típicos por la existencia de un atributo *retweeted_status*. Este atributo contiene una representación del Tweet original que fue retwiteado. Hay que tener en cuenta que los retwits del retwits no muestran representaciones del retwit intermediario, sino sólo del tweet original. *Ejemplo:*

- **source:** (*String*) Se utiliza para publicar el tweet en formato HTML. *Ejemplo:*

```
source: "<a href="http://twitter.com/download/iphone"
 rel="nofollow">Twitter for iPhone</a>"
```

- **text:** (*Nullable*) Texto UTF-8 de la actualización de estado.*Ejemplo:*

```
text: "Si la felicidad, como la gripe, es un estado
 q se transmite, contagia y propaga... Entonces causemos
 una epidemia!!"
```

- **user:** (*Users*) El usuario que publicó este Tweet. Pordemos observar sus atributos en las siguientes líneas de código:

```
"user":{"statuses_count":3080,
"favourites_count":22,
"protected":false,
"profile_text_color":"437792",
"profile_image_url":"...",
"name":"Twitter API",
"profile_sidebar_fill_color":"a9d9f1",
"listed_count":9252,
"following":true,
"profile_background_tile":false,
"utc_offset":-28800,
"description":"The Real Twitter API. I tweet about API
changes, service issues and happily answer questions
about Twitter and our API. Don't get an answer?
It's on my website.",
"location":"San Francisco, CA",
"contributors_enabled":true,
"verified":true,
"profile_link_color":"0094C2",
"followers_count":665829,
"url":"http:\/\/dev.twitter.com",
"default_profile":false,
"profile_sidebar_border_color":"0094C2",
"screen_name":"twitterapi",
"default_profile_image":false,
"notifications":false,
```

```
"display_url":null,
"show_all_inline_media":false,
"geo_enabled":true,
"profile_use_background_image":true,
"friends_count":32, "id_str":"6253282",
"entities":{"hashtags":[],
"urls":[],
"user_mentions":[]},
"expanded_url":null,
"is_translator":false,
"lang":"en",
"time_zone":"Pacific Time (US &amp;amp; Canada)",
"created_at":"Wed May 23 06:01:13 +0000 2007",
"profile_background_color":"e8f2f7",
"id":6253282,
"follow_request_sent":false,
"profile_background_image_url_https":"...",
"profile_background_image_url":"...",
"profile_image_url_https":"..."}
```


Finalmente, se añade una imagen que permite visualizar los campos de un Tweet:

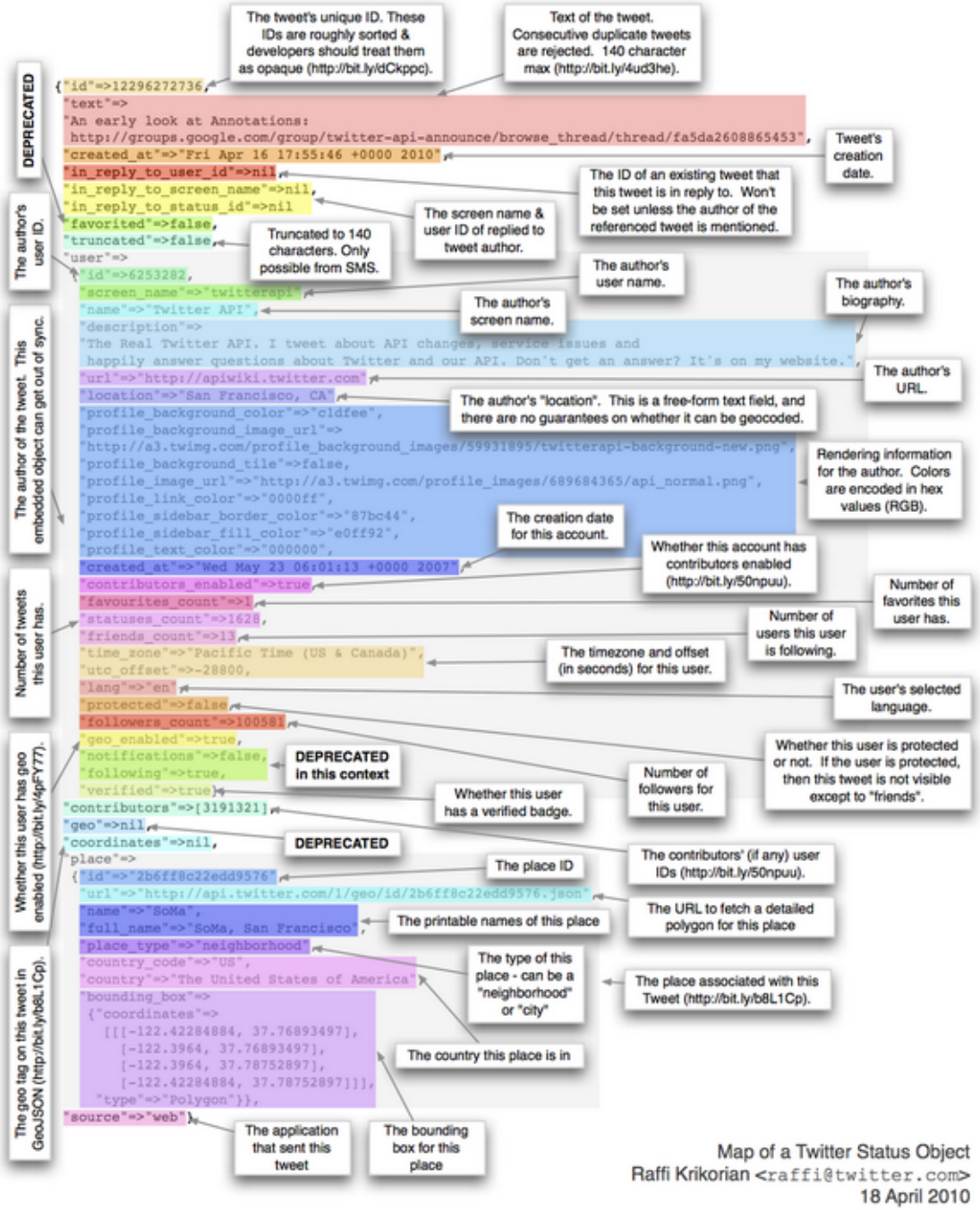


Figura 3.9: Descripción gráfica de la estructura de un Tweet [22].

4 Implementación

En esta sección se profundiza en la implementación del proyecto, y se divide en la siguientes secciones:

- **Infrastructura:** Breve descripción sobre el equipamiento necesario para el desarrollo de esta fase.
- **Flujo de datos:** Se detallan los procesos que transforman los datos del sistema, las distintas entidades que son fuente o destino de datos, de forma que se describe el movimiento de los mismos a través del sistema.
- **Estructura de datos:** Define la organización e interrelación de los datos y el conjunto de operaciones que se pueden realizar sobre ellos.

4.1. Infrastructura

Al comienzo de la fase de implementación, se realizó una instalación de un servidor en local, para empezar a desarrollar ciertas piezas básicas del proyecto. Para ello, se utilizó XAMPP, un servidor multiplataforma libre que consiste principalmente en el servidor web Apache, en la base de datos MySQL e intérpretes de los lenguajes Perl y PHP. Se eligió esta herramienta por su intuitiva instalación y rápido acceso a sus servicios.

En el momento de implementar el proceso de autenticación del usuario, se planteó la necesidad de hacer uso de un servidor remoto, ya que resultaba imposible para Twitter redirigir al usuario a la aplicación, una vez autenticado, si la dirección de retorno era una dirección local.

4.2. Flujo de datos

Mediante el siguiente diagrama de flujo se realiza una representación gráfica del proceso que sigue la aplicación desde su comienzo. Los pasos son los siguientes:

1. La aplicación comienza su ejecución en *index.php*. En donde comienza el proceso de autenticación OAuth.
2. Twitter recibe la petición de autenticación del usuario para utilizar la aplicación.

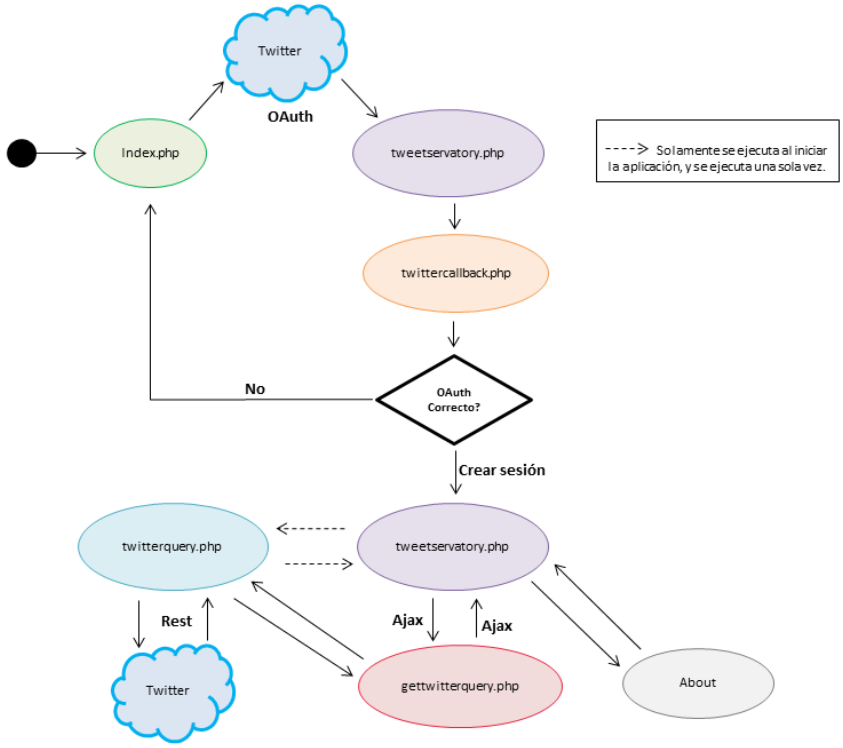


Figura 4.1: Diagrama de flujo de la aplicación.

3. Twitter redirige al usuario a la aplicación, es decir, a *Tweetservatory.php*.

4. Una vez en Tweetservatory, se inicia el código de la aplicación y se redirige a *twittercallback.php*, en donde se hace una comprobación de que todos los campos requeridos para la autenticación del usuario contienen la información correcta. Existen dos posibilidades:

   - La autenticación se desarrolla con normalidad y la aplicación sigue su curso.
   - Exite un error en la autenticación del usuario, en cuyo caso, se le redirige al comienzo del proceso, *index.php*.

5. La autenticación se ha desarrollado con éxito y se redirige al usuario a la aplicación, *tweetservatory.php*.

6. En la primera ejecución se dirige al usuario a *twitterquery.php*, en donde se encuentra una solicitud (query) preparada para ser enviada a Twitter como ejemplo de uso. Este paso se ejecuta solamente en al iniciarse la aplicación, y una sola vez. En el diagrama se distingue con flechas discontinúas. En este caso, se envia la query a Twitter, se obtiene una respuesta JSON de Twitter en el mismo fichero, y se reenvían estos datos a la aplicación para que sea posible procesarlos y mostrarlos.

7. A partir de este momento, el usuario puede comenzar a realizar peticiones de datos

a Twitter. Cada vez que el usuario realice una solicitud, mediante el módulo de AJAX se redirige al usuario al fichero *gettwitterquery.php*. Este fichero tiene como función enviar los datos de la query creada a *twitterquery.php*. Este finchero tiene la misma labor descrita en anteriores pasos, envía la solicitud a Twitter. La información JSON de respuesta de Twitter se devuelve a *gettwitterquery.php*, de forma que pueda ser reenviada a la aplicación principal (*tweetservatory.php*) para poder procesar la información y mostrarla al usuario.

8. Por último existe otra posibilidad, y es *About*. En donde se realiza una descripción sobre el uso de la aplicación, sus casos de uso y se ofrece información básica del autor.

4.3. Estructura de datos

4.3.1. Tipo abstracto de datos

En este proyecto se utiliza la programación orientada a objetos por los siguientes motivos principales:

- Agiliza y facilita el desarrollo.
- Se crean sistemas mas flexibles en los que al manipular las distintas clases, cualquier cambio que se realice sobre ellas quedará reflejado automáticamente en cualquier lugar en donde aparezcan.
- Facilita la comprensión de la estructura de datos.
- Facilita la ampliación del proyecto.

El objeto principal de esta aplicación es **Term**, que se asocia al término buscado por el usuario en cada una de sus solicitudes de datos. Al poderse hacer tantas solicitudes como Twitter permita dentro de sus límites, se va guardando la información de cada una de estas peticiones (mientras dure la sesión del usuario).

La datos asociados a cada objeto Term, se guardan dentro del objeto. De esta forma se tiene un acceso rápido y más sencillo a esos atributos que si se utilizasen distintos arrays para guardar los diferentes datos.

El conjunto de términos se almacenan en una **tabla hash** con el nombre de **terms**. Esta estructura de datos asocia claves con valores. Una de sus mayores ventajas es que soporta búsquedas de manera eficiente, permitiendo el acceso a los elementos que se almacenan según la clave generada. Los elementos de esta tabla hash se organizan la siguiente forma:

- **clave:** Etiqueta (cadena de caracteres) que identifica a cada término.
- **valor:** El objeto que corresponde al término (Term).

A su vez, la clase Term recoge en sus atributos distintos tipos de información:

- Existe un tipo de información *preprocesada* que tiene como fin agilizar el proceso de mostrar resultados al usuario y evitar trabajo extra a la aplicación. Implica que se guardan ciertos datos al procesar los resultados obtenidos de Twitter (JSON), para así, no tener que volver a procesar el fichero JSON cada vez que se quiera tener acceso a esa información. Este tipo de datos son los atributos de la clase Preprocessed, que contiene únicamente información que requiere el procesamiento entero del fichero JSON devuelto por TWitter, lo cual supondría un retardo considerable a la hora de volver a disponer de estos datos.
    - **tweetslist** contiene la lista completa de Tweets asociada al término.
    - **tweetsgeolist** contiene la lista de Tweets geolocalizados asociada al término.
    - **markers** es la lista de marcadores de Tweets geolocalizados asociada al término.
    - **Chartjs** es un objeto que tiene como atributos las gráficas de datos a mostrar.Estas son: La gráfica estadística, y las dos gráficas que muestran el número de Tweets y el grado de los usuarios que utilizan el término en función del tiempo.
- La información sin procesar o *Raw Data*, se almacena con el fin de tener disposición de los datos en el caso de que fuera necesario. A su vez, permite el acceso a la información de los Tweets que no se estudia en este proyecto, pero que permite que en futuras versiones se tenga diponible.
    - Fichero **JSON** que se obtiene de Twitter en respuesta a una petición.
- El resto de atributos no comparte las características de los otros dos tipos de datos, y hacen referencia al resto de datos relevantes sobre el término. Son los siguientes:
    - El atributo **tag** es la etiqueta de identifica al término.
    - El atributo **color** asocia un color a un término, de forma que ese término, sus marcadores y las gráficas sociadas a él, mantienen este color.
    - El **objeto Cont** tiene como atributos los contadores necesarios para recopilar cifras asociadas a los Tweets del término. Estos son:
        - **cont** = contador del número total de Tweets que se está analizando.

  - **contegeo** = contador del número de Tweets geolocalizados que se está analizando.
  - **urls** = contador de URLs que posee el conjunto de Tweets analizado.
  - **rt** = contador de Retweets que posee el conjunto de Tweets analizado.
  - **hashtags** = contador de hashtags que posee el conjunto de Tweets analizado.
  - **mentions** = contador de menciones que posee el conjunto de Tweets analizado.
- El **objeto Query** tiene como atributos aquellos campos necesarios para poder solicitar a Twitter datos pasados o más actuales del término. Estos campos son:
  - **next** = query para solicitar los 100 Tweets anteriores en el tiempo asociados al término.
  - **refresh** = query para solicitar Tweets más actuales que los Tweets ofrecidos que se asocian al término.
  - **max_id** = Número de referencia para controlar el identificador máximo de los Tweets obtenidos.
- El **objeto TimeData** posee los atributos relacionados con tiempos que permiten establecer un rango de tiempos para hacer posible la visualización de datos a lo largo del tiempo.
  - **timeTweets** = Lista del número de Tweets asociados al término en los distintos tiempos a representar.
  - **timeDegree** = Lista del grado de los usuarios asociados al término en los distintos tiempos a representar.
  - **mintime** = Mínimo de los tiempos a representar en las gráficas.
  - **maxtime** = Máximo de los tiempos a representar en las gráficas.
- El **objeto Top** tiene como atributos las listas de los elementos más populares estudiados.
  - **usersLst** = tabla hash de usuarios según su grado de difusión de información del término dentro del rango de Tweets estudiado. Su clave es el nombre del usuario, y el valor, su grado.
  - **rtLst** = tabla hash de Tweets según el número de Retweets que se asocian al término dentro del rango de Tweets estudiado. Su clave es el Tweet retuiteado, y el valor, el número de veces que aparece.
  - **hstgLst** = tabla hash de hashtags seún su popularidad que se asocian al término dentro del rango de Tweets estudiado. Su clave es el hashtag, y el valor, el número de veces que aparece.

- **urlLst** = tabla hash de URLs según su uso en lo Tweets estudiados asociados al término. Su clave es una URL, y el valor, el número de veces que aparece.
- **mtLst** = tabla hash de menciones según su popularidad en los Tweets estudiados asociados al término. Su clave es el campo mención, y el valor, el número de veces que aparece.

El diagrama de clases asociado a esta estructura de datos es el siguiente:

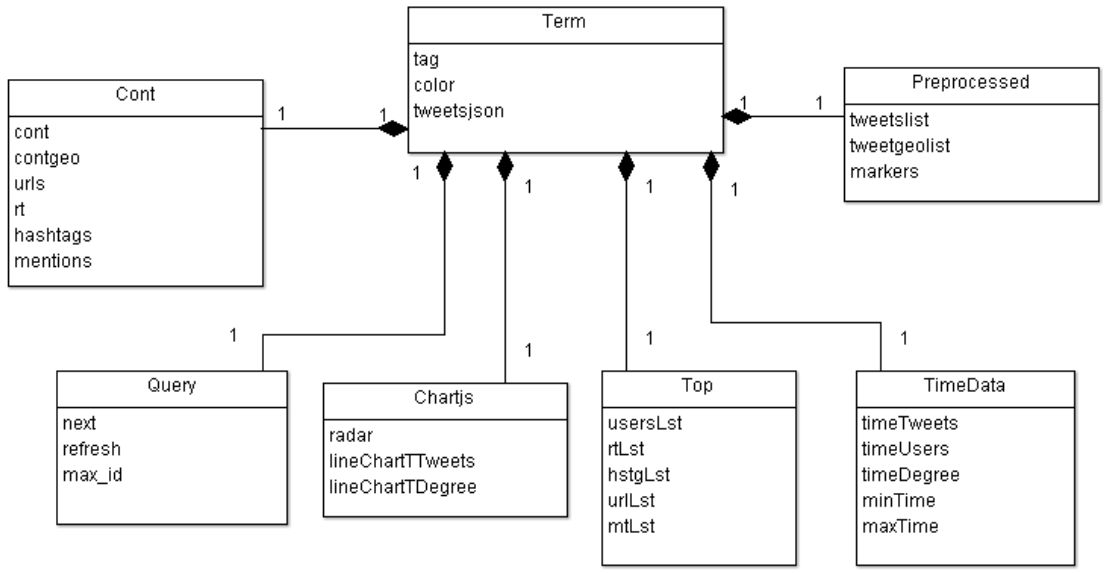


Figura 4.2: Diagrama de clases de la aplicación.

Se utiliza una organización de clases sencilla, en la que cada una de las clases poseen atributos que tienen relación unos con otros. Se podría organizar de forma que la única clase que formase la estructura de datos fuera la clase Term, pero se hace esta división por clases para tener una estructura más clara y ordenada.

Se utiliza la relación de composición para definir la relación entre la clase Term con el resto de clases, ya que Term está compuesta por estas clases. En esta relación, la vida de la clase contenida depende totalmente de la vida de la clase contenedor, es decir, si la clase contenedor se elimina, también se eliminan las clases contenidas.

4.3.2. Procesado de información

Una vez retornados los resultados de la solicitud a Twitter mediante AJAX, se realiza el procesado de los mismos.

Procesado de metadatos

Este proceso se lleva a cabo en la función *processTwitterQuery*, imprescindible para el funcionamiento de la aplicación. Se comienza parseando el fichero JSON devuelto como resultado de la consulta realizada a Twitter, de forma que se pueda acceder a la información deseada.

En este punto se extraen y almacenan los datos que se requieren para ofrecer las distintas funcionalidades. Estos datos son: *query*, *next_ results*, *refresh_ url* y *max_ id*.

Procesado de Tweets

Para comenzar el procesado de Tweets se utiliza la función *processTwitterQuery*, que además de hace el procesado de metadatos. En base a los resultados obtenidos en el procesado de metadatos y en función de si ya existía el término (y si es una búsqueda hacia atrás o hacia adelante en el tiempo) o si es uno nuevo, se actualizan los valores generales del término o se crea uno nuevo.

A continuación, *processTweets* se encarga de procesar los Tweets uno a uno y obtener la información precisa de cada uno de ellos. A continuación, se estructura la sección en distintas subsecciones para describir el procesado de datos que se sigue hasta alcanzar las funcionalidades expuestas.

Creación de marcadores geolocalizados: En la función *processTweets* se hace una comprobación de si cada uno de los Tweets está geolocalizado. En caso afirmativo, se añade el marcador a la lista del marcadores asociada al término mediante la función *addMarker*,y se añade el Tweet a la lista de Tweets geolocalizados.

En este momento, el marcador todavía no ha sido creado, de tal manera que en *addMArker* se deben preparar los distintos datos que se desean mostrar en un marcador para que posteriormente se obtenga una visualización correcta de éste, crear el marcador y finalmente, añadirlo a la lista de marcadores.

Creación de listas de Tweets: Como se ha comentado anteriormente, el procesado de cada uno de los Tweets se realiza en la función en la función *processTweets*. Es aquí donde los Tweets se añaden a la lista de Tweets generales del término y en caso de estar geolocalizados, se añaden a la lista de Tweets geolocalizados del término.

Cómputo de estadísticas generales: A su vez, es en la función *processTweets* donde se obtienen los datos estadísticos que posteriormente se mostrarán de manera gráfica utilizando la librería “chart.js".

Estos datos se obtienen en el procesado de cada Tweet, donde se comprueba el número de URLs, hashtags, menciones de usuarios y Retweets que contiene el Tweet.

Según los datos que se obtengan en este proceso, se actualizan los distintos atributos del término asociados a estos campos. Además de estos atributos, se actualizan los contadores de la lista de Tweets general y la lista de Tweets geolocalizados asociada al término.

Una vez se poseen todos los datos, se llama a la función *generateRadarChart*, que general la gráfica estadística. Finalmente, mediante la función *updateInfoPanel*, se imprime la estadística, ya que esta función actualiza los contenidos mostrados en panel de información completo.

Cálculo de estadíticas temporales: En este caso,*processTweets* es la función en la que se obtienen los datos a mostrar en las gráficas.

Los datos a representar en la estadítica de **Tweets en función del tiempo** se consiguen en un proceso en el que se comienza analizando la fecha de creación de cada Tweet. Para tratar esta fecha, se transforma en horas, ya que es la unidad de división de tiempos que se pretende utilizar en la gráfica.

Para establecer los límites de acotación de la función a mostrar, se comprueba si la fecha de cada uno de los Tweets es menor a la mínima fecha almacenada en el objeto timeData del término (minTime) o mayor a la fecha máxima almacenada en el objeto timeData del término (maxTime), de forma que si se cumple alguna de esta condiciones, se actualizan los atributos pertinentes.

Una vez establecidos los límites de acotación, se comprueba si la fecha de cada uno de los Tweets ya ha sido guardada en la lista de tiempos (timeTweets) del objeto timeData del término. En caso afirmativo, se aumenta el contador de la posición de esa fecha concreta. En caso contrario, se crea una nueva posición en la lista para era fecha y se inicializa su valor a 1.

Los datos a representar en la estadítica de **Grado de usuarios en función del tiempo** se consiguen haciendo una llamada a la función *getTimeDegree*, en la que se comprueba si en la lista de tiempos (timeDegree) del objeto timeData del término está vacía o no. En el caso de estar vacía, se añade el grado del usuario del Tweets que se está analizando. En caso de no estar vacía, se actualiza esa posición con el grado de los usuarios que han tuiteado en esa fecha.

Una vez que se han almacenado los datos a mostrar en ambas gráficas, se llama a la función *generateLineChart* para generar cada una de las gráficas. Esta función genera una de las gráficas u otra dependiendo de si se le hace una llamada con el parámetro "type.ª 1 (generar gráfica de grados de usuario en función del tiempo) o a 0 (generar gráfica de Tweets en función del tiempo). Para generar estas gráficas se utilizan las función *getChartLabels* y *getChartData*.

getChartLabels es una función válida para ambos casos, ya que se encarga de hacer divisiones en el tiempo dentro de los límites calculados y mostrar el formato de esas divisiones en horas por día (labels).

Sin embargo,*getChartData* no puede ser igual para mostrar las dos gráficas, ya que se muestran datos distintos. Por ello, se vuelve a utilizar el campo “type"descrito anteriormente para diferenciar las dos llamadas. En el caso de que “type"sea 0, se rellena el array de datos a mostrar con los datos del array *timeTweets*. Sin embargo, cuando “type"tenga el valor 1, se rellenará el array de datos a mostrar con los datos del array *timeDegree*.

Finalmente, igual que en el caso de las estadísticas, mediante la función *updateInfoPanel*, se imprimen las distintas gráficas actualizadas, al actualizar el panel de datos al completo.

Listas destacadas: Los datos de estas listas se obtienen en la función *processTweets*. En este caso, para todos los Tweets analizados, se van rellenando las tablas hash del objeto Top del término (usersLst, rtLst, hstgLst, urlLst y mtLst) según su clave y el número de veces que se dan las caracteríticas de cada una de ellas en los Tweets.

A continuación, se hace una llamada a la función *generateTopLists*, que ordena las listas en función de los mayores valores y las acota a un máximo de 5 valores.

Por último, realizando una llamada a la función *updateInfoPanel*, se actualizarán las distintas listas del panel de visualización de datos.

Tareas de apoyo: Para la realización de tareas de apoyo como asignar un color a un término, colorear los términos de un texto o extraer la fecha en un formato específico, se han creado una serie de funciones de apoyo. La documentación técnica de todas estas funciones del proyecto, puede encontrarse en [30].

5 Casos de uso

5.1. Descripción general de la interfaz

En esta sección se va a describir el uso general de la aplicación detallando cada uno de los elementos de la interfaz.

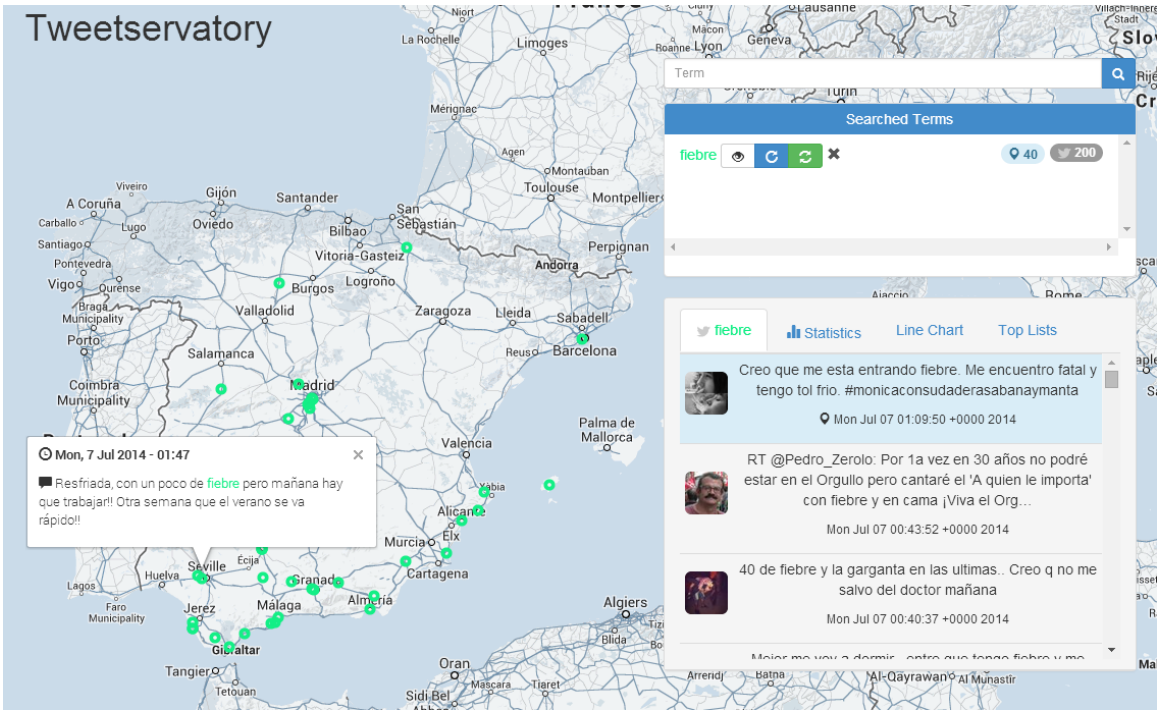


Figura 5.1: Visión general de la aplicación.

El elemento principal de la aplicación es el **mapa**, el cual es el elemento que ofrece un mayor rango de información. El mapa ocupa el espacio completo de la pantalla, de forma que se pueda obtener el máximo partido de él. La información que muestra el mapa consiste en distintos marcadores que identifican Tweets geolocalizados. Estos marcadores se pueden desplegar, de forma que se muestre la información concreta del Tweet.

Mediante el botón del icono, cuya imagen es un ojo, se puede elegir visualizar u ocultar los marcadores. Esto se puede observar en la figura 5.2.

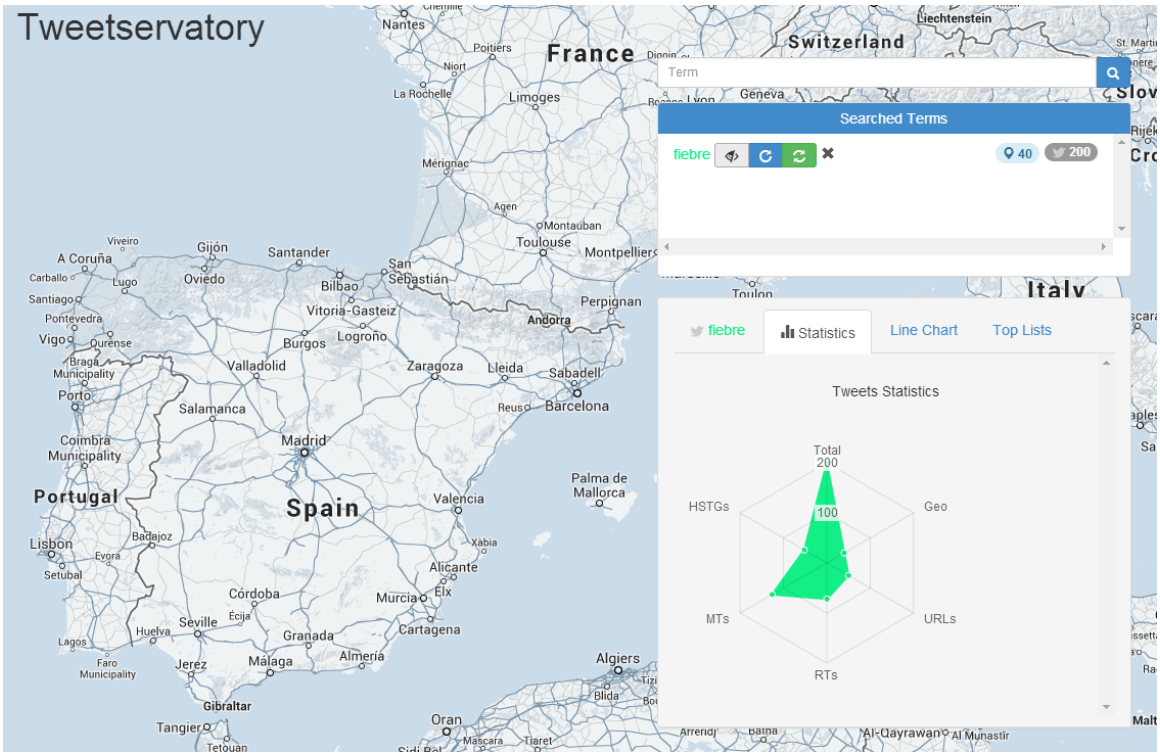


Figura 5.2: Visualizar u ocultar los marcadores asociados a un término.

Al realizar la búsqueda de un término, éste se añadirá en la **lista de terminos** (ver figura 5.3).

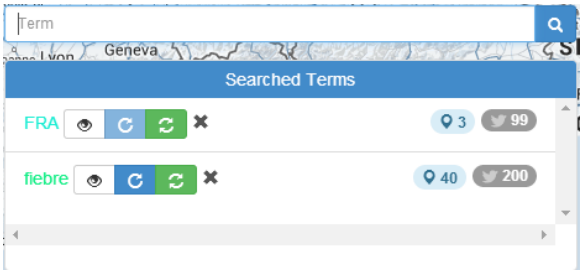
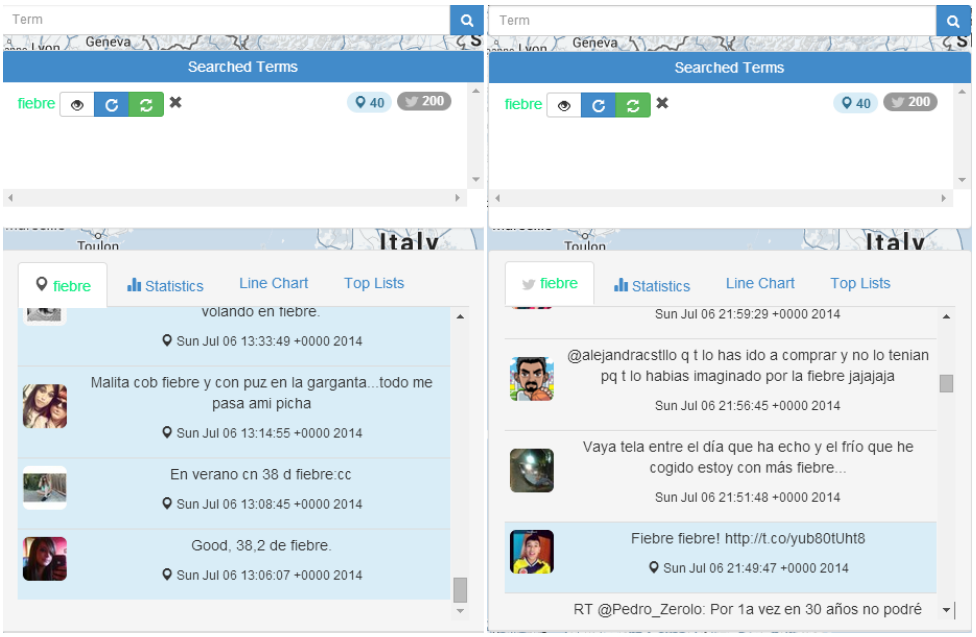


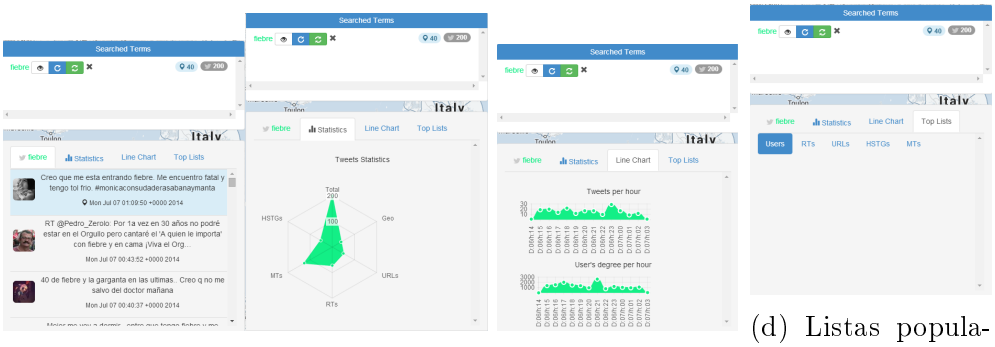
Figura 5.3: Lista de términos

Una vez se ha añadido el término a la lista, se puede disponer de la lista de todos los Tweets haciendo click en el botón gris que contiene el icono de Twitter, y de la lista de los Tweets geolocalizados, haciendo click en en el botón azul que contiene el icono de geolozalización. Los Tweets geolocalizados pueden distinguirse en las listas por tu color azul de fondo y por su icono de geolocalización (ver figura). Además, es posible eliminar el término de la lista pulsando el botón que contiene una cruz, añadir Tweets más recientes a la lista pulsando el botón verde y añadir los 100 Tweets anteriores en el tiempo pulsando el botón azul.



(a) Lista de Tweets geolocalizados. (b) Lista de Tweets general.

Figura 5.4: Listas de Tweets



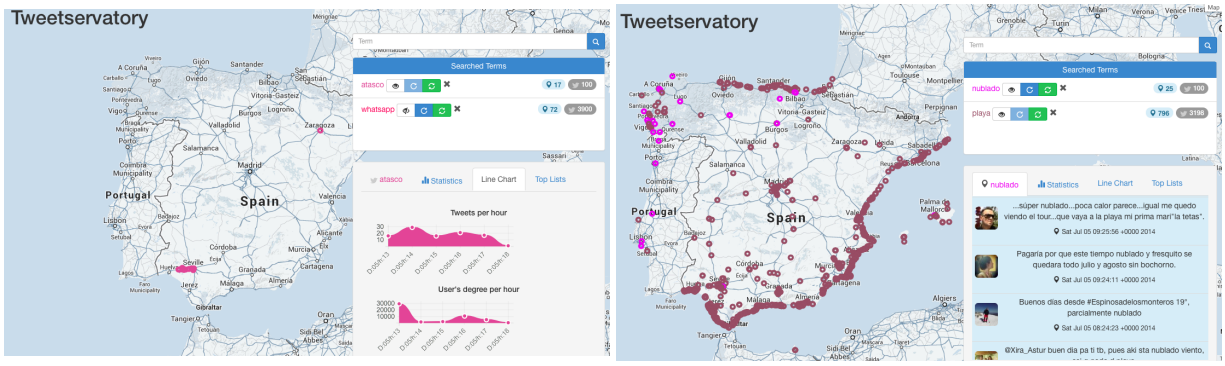
(a) Tweets (b) Estadísticas (c) Linea temporal (d) Listas populares

Figura 5.5: Distintas pestañas del panel de datos.

El **panel de datos** situado debajo de las lista de términos se forma por 4 pestañas. La primera muestra la lista de Tweets (ya sea la lista general o la lista de Tweets geolocalizados), la segunda muestra las estadísticas generales de los Tweets, la tercera muestra las gráficas de número de Tweets y grado de usuarios en función del tiempo, y por último, la cuarta pestaña muestra las distintas listas de elementos con más popularidad de ciertos campos significativos del Tweet. Se pueden visualizar en la figura 5.5.

5.2. Casos de uso de las herramientas de visualización

Se han realizado una serie de estudios de uso que permitan comprender la aportación de información de la **geolocalización**. En las siguiente imágenes se puede apreciar como en la búsqueda del término "atasco" se distingue claramente uno de los atascos formados



(a) Visualización del atasco en la operación salida de principios de julio de 2014.

(b) Comparación de las búsquedas "playa" y "nublado".

Figura 5.6: Figuras sobre clima y tráfico.

en la operación salida de principios de julio del 2014. Además, en el caso de la búsqueda de los términos "playa" y "nublado", se pueden distinguir las parte de España en donde se da cada una de esas situaciones.

Estos ejemplos ilustran como la información geográfica refleja de forma clara, información sobre el tráfico y el clima. Para concretar más, en la figura 5.7 se analizan las siguientes imágenes en las que se busca el término "hurrican" pocas horas antes de que llegue a Carolina del Norte. Esta información nos permite ratificar, que la geolocalización detalla claramente fenómenos meteorológicos.

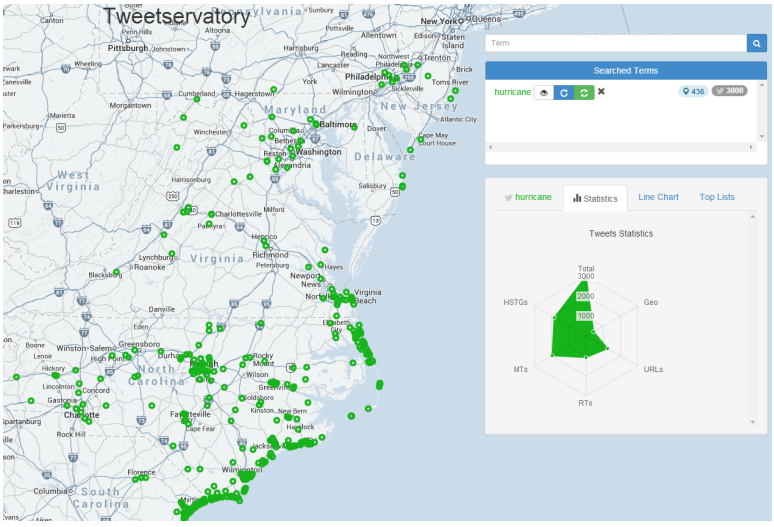


Figura 5.7: Huracán Arthur unas horas antes de pasar por Carolina del Norte.

Asimismo, otros estudios que se pueden hacer son aquellos que tienen que ver con factores migratorios y comunidades lingüísticas. En la figura 5.8 se observa claramente como las búsquedas de los términos "rapa", "chiquillo" y "zagal" situan distintas comunidades lingüísticas en el mapa, de forma que se aprecian los lugares en los que más se utilizan

este tipo de términos.

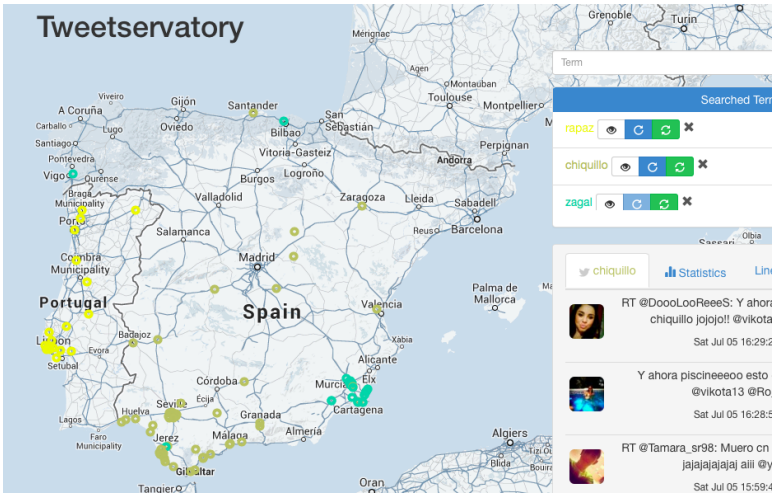
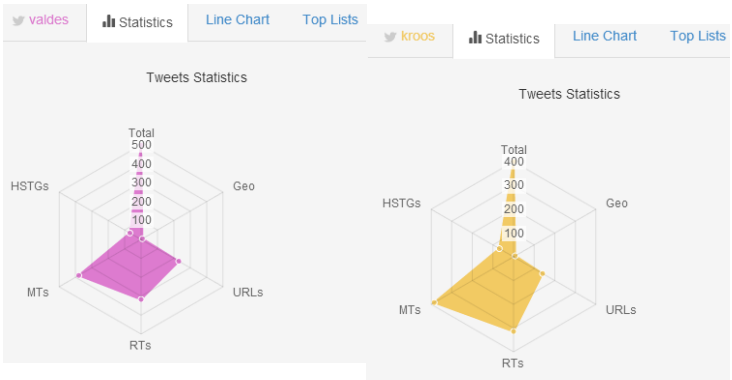


Figura 5.8: Comunidades lingüísticas.

Con la visualización gráfica de estadísticas, se logra una caracterización de términos. La reducción de dimensionalidad y la representación gráfica son aspectos fundamentales para localizar distintas huellas digitales. Un ejemplo que muestra este tipo de clases es el mostrado en la figura 5.9, en la cual se comparan las firmas de dos jugadores de fútbol. Como se puede apreciar, resultan muy parecidas y permiten identificar elementos que están relacionados.

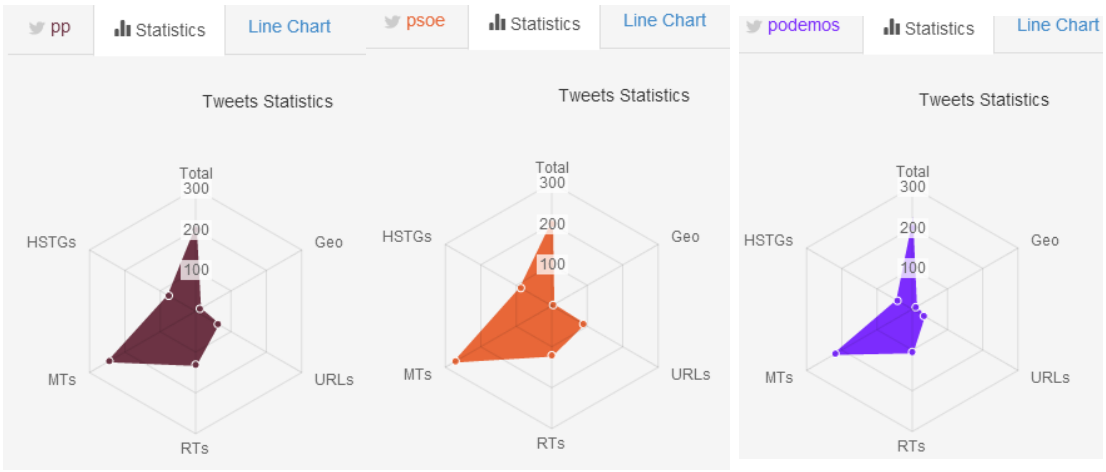
Existen otros ámbitos en los que también se consiguen este tipo de resultados. Los eventos relacionados suelen tener en común gran parte de sus atributos, permitiendo realizar una clasificación de eventos. A continuación se muestran las huellas digitales de tres partidos políticos. Los partidos “pp” y “psoe” son partidos políticos que se establecieron en España hace un tiempo relativamente. Según la figura 5.10, se puede llegar a la conclusión, de que el “podemos” ha encontrado un lugar dentro de los partidos políticos de



(a) Jugador de fútbol: Valdés.

(b) Jugador de fútbol: Kross.

Figura 5.9: Comparación de huellas digitales de dos jugadores de fútbol.



(a) Partido político pp. (b) Partido político psoe (c) Partido político podemos.

Figura 5.10: Comparación de huellas digitales de partidos políticos.

España, ya que en estas fechas posee una clase que se corresponde con a que identifica partidos políticos.

La representación del grado de los usuarios en función del tiempo tiene como fin principal actuar como sensor. De esta manera se predice la viralidad, es decir, es muy probable que una subida del grado de los usuarios vaya seguida de una subida en el número de usos. Por ejemplo, eventos que reciben una atención mediática, suelen mostrar posteriormente una subida en el número de usos. Este caso, se confirma esta teoría utilizando el ejemplo de una granizada que se dió el 3 de julio de 2014. Se realiza una difusión del evento por usuarios que tienen un alto grado de seguidores, como son Iberia, InfoEmerg y La Razón.

@larazon_ es: Estamos recopilando fotos de la #granizada sobre Barajas y otros puntos de España. Envíanos la tuya y la publicaremos en la...

Gracias @InfoEmerg por ayudarnos #haciendoProtecciónCivil compartiendo la información durante la #granizada de #Mejorada. Un fuerte abrazo!

Iberia: @javi1773 Efectivamente, la granizada que ha caído en Madrid ha afectado al aeropuerto...

A continuación, en la figura 5.12, se observa una subida del uso del término "granizada" en la representación del uso de Tweets por hora.

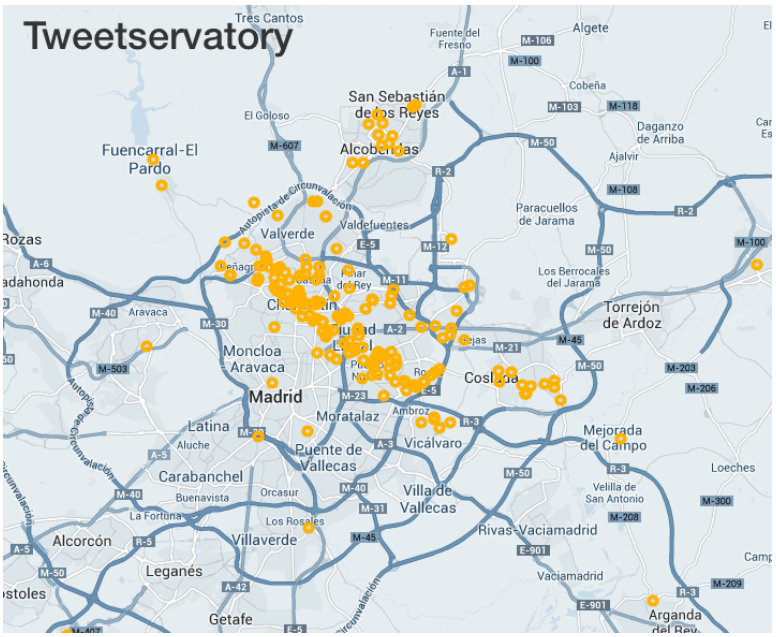


Figura 5.11: Visualización del lugar de la granizada.

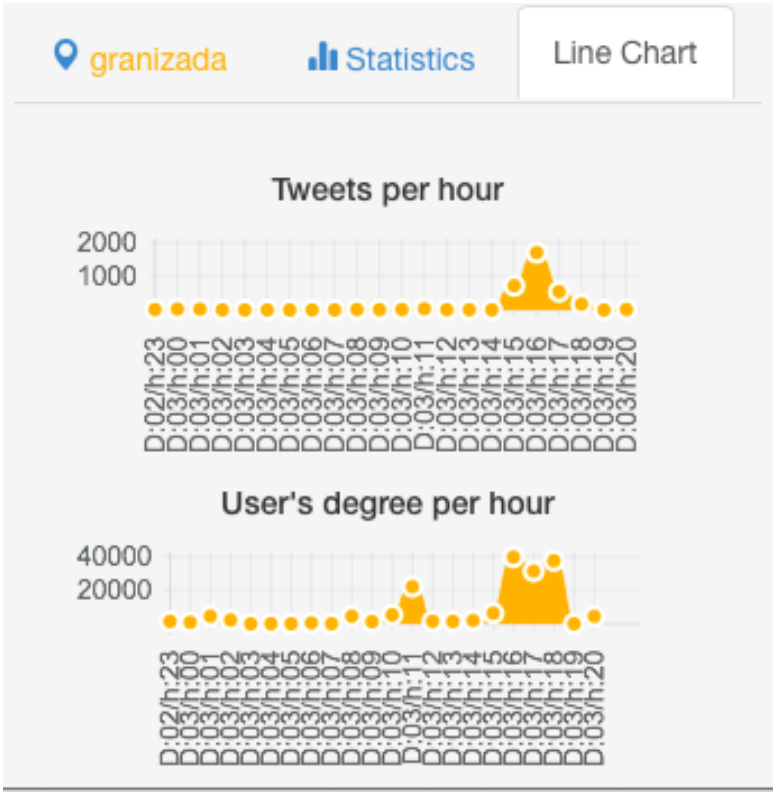


Figura 5.12: Picos en los grados de los usuarios y en el uso del término “granizada” en los Tweets procesados.

6 Conclusiones

El desarrollo de esta herramienta ha requerido una gran cantidad de pruebas y un fuerte uso. Según en análisis inicial del estado del arte 2 como experiencia de uso, se han obtenido ciertas conclusiones destacadas. En la construcción de esta aplicación de visualización interactiva de Twitter, se trata con distintos tipos de variables. Estas variables son las que proporcionan una herramienta para llegar a hacer un análisis profundo con menos esfuerzo.

Aquellas variables que proporcionan más información al usuario son el mapa y las estadísticas. El mapa es una forma muy rápida de obtener mucha información, de ahí la importancia de la geolocalización. A su vez, las estadísticas ofrecen información muy en detalle respecto a algo, mientras que posibilita la diferenciación de clases.

Sin embargo, la evolución temporal del grado y las listas del "top de elementos" aportan menos información comparándolas con las citadas anteriormente. La evolución temporal del grado del usuario es muy simple y mejorable, pero incluso con esa simplicidad, le permite al usuario averiguar si un evento se ha trascendido a Twitter, además de estimar el impacto futuro de un determinado término.

Las listas de los atributos del Tweet más populares permiten al usuario ampliar su radio de búsqueda, es decir, expandir su búsqueda. Igualmente, permite transformar el análisis que se está realizando en una acción como usuario de Twitter, como por ejemplo, seguir a un usuario.

Con esto, se concluye que los mapas y los datos estadísticos ofrecen un mayor porcentaje de información. Se puede considerar que es una información más básica (80 % aproximadamente). Además, se considera que la linea temporal del grado de los usuarios ofrece un porcentaje menor de información (15 % aproximadamente). Y finalmente, la lista de elementos del Tweet más populares relacionados con un término, ofrecen un porcentaje aún menor de información útil (5 % aproximadamente).

7 Trabajo futuro

Con el fin de ampliar las funcionalidades de la herramienta en un futuro, se plantear aspectos mejorables y que puedan hacer nuevas aportaciones al proyecto:

Descarga y carga de archivos JSON: Una de la opciones más atrayentes para realizar investigaciones es añadir la opción de cargar ficheros de datos JSON a la aplicación, con el fin de poder analizar datos que no dependan de las devoluciones de información de Twitter y de sus limitaciones. Además, resultaría muy útil el poder descargar archivos JSON de forma que si se encuentran datos interesantes, no se pierdan y se puedan volver a analizar en otro momento.

Filtrado de información en base al tiempo: Esta funcionalidad permitiría analizar ciertos datos que están acotados entre dos fechas cualesquiera, es decir, se podría ofrecer una acotación personalizada.

Filtrado de búsquedas que ya han sido hechas: Con ello se pretende hacer nuevos filtrados en base a unos resultados concretos.

Restricciones del Zoom: Twitter posee ciertas restricciones, y es que, aunque esta aplicación adapta el radio de búsqueda de una query al zoom del mapa, Twitter tiene limitaciones respecto a ese espacio de búsqueda. Llega un punto en el que, aunque el radio sea el correcto respecto al nivel de zoom, Twitter no ofrece la información para todo ese espacio. Se han realizado comprobaciones en las que a un nivel de zoom grande (lejano), Norte América se divide en 3 zonas, la costa este, la cosa oeste y el centro, y devuelve datos conforme a esa división.

Mejora de la interfaz: Hay ciertos aspectos mejorables de la interfaz, como son mostrar más datos en la estadística para limitar menos las huellas digitales o mostrar otros campos no utilizados del Tweet.

Visualización de Tweets: Hasta ahora se pueden estudiar las listas de Tweets generales y geolocalizados, pero cuando se visualiza la lista de Tweets geolocalizados, resulta complicado, sobretodo cuando existen muchos Tweets geolocalizados, encontrar un Tweet de la lista en el mapa. Probablemente, pueda ser una característica interesante el poder hacer click sobre uno de los Tweets geolocalizados y que seguidamente, se abra la información asociada a ese Tweet en el mapa.

Adaptación del sistema a tablets: Utilizando Boostrap es sencillo adaptar el tamaño y la distribución de aplicaciones a distintos dispositivos móviles de diferente tamaño. Se utiliza Bootstrap para la organización de los elementos de la aplicación, pero no se contemplan los distintos tamaños de pantalla.

Bibliografía

[1] Twitter Usage Statistics. http://www.internetlivestats.com/twitter-statistics/#ref-2, 2013. [Online; accessed 2014].

[2] Eiji Aramaki, Sachiko Maskawa, and Mizuki Morita. Twitter catches the flu: detecting influenza epidemics using twitter. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1568–1576. Association for Computational Linguistics, 2011.

[3] Eytan Bakshy, Jake M Hofman, Winter A Mason, and Duncan J Watts. Everyone's an influencer: quantifying influence on twitter. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 65–74. ACM, 2011.

[4] John Barratt. Trendsmap. http://trendsmap.com/, 2013. [Online; accessed 2014].

[5] TweetBeam B.V. Tweet Beam. http://www.tweetbeam.com/. [Online; accessed 2014].

[6] Yang Cai, Rafael de M Franco, and Manuel García-Herranz. Visual latency-based interactive visualization for digital forensics. *Journal of Computational Science*, 1(2):115–120, 2010.

[7] Riley Crane and Didier Sornette. Robust dynamic classes revealed by measuring the response function of a social system. *Proceedings of the National Academy of Sciences*, 105(41):15649–15653, 2008.

[8] Varios desarrolladores. MentionApp. http://mentionmapp.com/. [Online; accessed 2014].

[9] Varios desarrolladores. Revisit. http://moritz.stefaner.eu/projects/revisit/. [Online; accessed 2014].

[10] Varios desarrolladores. Sentiment Viz. http://www.csc.ncsu.edu/faculty/healey/tweet_viz/tweet_app/. [Online; accessed 2014].

[11] Varios desarrolladores. Twitter Stream Graphs. http://www.neoformix.com/Projects/TwitterStreamGraphs/view.php, 2009. [Online; accessed 2014].

[12] Varios desarrolladores. Tweet Archivist. https://www.tweetarchivist.com/, 2014. [Online; accessed 2014].

[13] Google Developers. Documentación de la versión 3 del API de JavaScript de Google Maps. https://developers.google.com/maps/documentation/javascript/tutorial, 2013. [Online; accessed 2014].

[14] Twitter Developers. Diferencias entre REST API y Streaming API. https://dev.twitter.com/docs/api/streaming, 2012. [Online; accessed 2014].

[15] Twitter Developers. Diferentes tipos de limitación de tasa de queries. https://dev.twitter.com/docs/rate-limiting/1.1/limits, 2013. [Online; accessed 2014].

[16] Twitter Developers. Limitación de tasa de queries. https://dev.twitter.com/docs/rate-limiting/1.1, 2013. [Online; accessed 2014].

[17] Twitter Developers. Parámetros que forman un Tweet. https://dev.twitter.com/docs/platform-objects/tweets, 2013. [Online; accessed 2014].

[18] Twitter Developers. Proceso de la autorización 3-legged. https://dev.twitter.com/docs/auth/implementing-sign-twitter, 2013. [Online; accessed 2014].

[19] Twitter Developers. Estructura de una query. https://dev.twitter.com/docs/api/1.1/get/search/tweets, 2014. [Online; accessed 2014].

[20] Manuel Garcia-Herranz, Esteban Moro, Manuel Cebrian, Nicholas A Christakis, and James H Fowler. Using friends as sensors to detect global-scale contagious outbreaks. *PloS one*, 9(4):e92413, 2014.

[21] Sandra González-Bailón, Javier Borge-Holthoefer, and Yamir Moreno. Broadcasters and hidden influentials in online protest diffusion. *American Behavioral Scientist*, page 0002764213479371, 2013.

[22] Raffi Krikorian. Map of a Tweet. https://twitter.com/tmccormick/status/435461469527097345/photo/1, 2010. [Online; accessed 2014].

[23] Raffi Krikorian. New Tweets per second record, and how! https://blog.twitter.com/2013/new-tweets-per-second-record-and-how, 2013. [Online; accessed 2014].

[24] Yury Kryvasheyeu, Haohui Chen, Esteban Moro, Pascal Van Hentenryck, and Manuel Cebrian. Performance of social network sensors during hurricane sandy. *arXiv preprint arXiv:1402.2482*, 2014.

[25] Janette Lehmann, Bruno Gonçalves, José J Ramasco, and Ciro Cattuto. Dynamical classes of collective attention in twitter. In *Proceedings of the 21st international conference on World Wide Web*, pages 251–260. ACM, 2012.

[26] Gilad Lotan, Erhardt Graeff, Mike Ananny, Devin Gaffney, Ian Pearce, et al. The arab spring| the revolutions were tweeted: Information flows during the 2011 tunisian and egyptian revolutions. *International Journal of Communication*, 5:31, 2011.

[27] CyBranding Ltd. Hashtagify. http://hashtagify.me/, 2012. [Online; accessed 2014].

[28] Greg Miller. Social scientists wade into the tweet stream. *Science*, 333(6051):1814–1815, 2011.

[29] Aída Morcillo Rincón. About. https://dev.twitter.com/, 2014. [Online; accessed 2014].

[30] Aída Morcillo Rincón. About. https://tweetservatory/current/about, 2014. [Online; accessed 2014].

[31] Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. Earthquake shakes twitter users: real-time event detection by social sensors. In *Proceedings of the 19th international conference on World wide web*, pages 851–860. ACM, 2010.

[32] StudioIMC and Web5design. Twitt3d. http://www.twitt3d.com/, 2010. [Online; accessed 2014].

[33] Boris Veldhuijzen van Zanten. Twitter Counter. http://twittercounter.com/, 2008. [Online; accessed 2014].

[34] Kevin Weil. Measuring Tweets. https://blog.twitter.com/2010/measuring-tweets, 2010. [Online; accessed 2014].

[35] Sharon Weinberger. Spies to use twitter as crystal ball. *Nature*, 478(7369), 2011.

[36] Lance Whitney. Twitter user growth rates to continue to fall. http://www.cnet.com/news/twitter-user-growth-rates-to-continue-to-fall-report/, 2014. [Online; accessed 2014].

[37] Jaewon Yang and Jure Leskovec. Patterns of temporal variation in online media. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 177–186. ACM, 2011.

A Estructura de un Tweet

A continuación se realiza una descripción detallada sobre los parámetros de un Tweet:

- **contributors:** (*Collection of contributors; Nullable*) Una colección de objetos de usuario, breves, (por lo general solamente uno) que indica los usuarios que contribuyeron a la autoría del tweet, en nombre del tweet oficial. *Ejemplo:*

```
"contributors":
[
    {
        "id":819797,
        "id_str":"819797",
        "screen_name":"episod"
    }
]
```

- **coordinates:** (*Coordinates; Nullable*)Representa la ubicación geográfica del Tweet según lo informado por el usuario o la aplicación cliente. La matriz de coordenadas tiene un formato en el que se sitúa primero la longitud, y a continuación la latitud. *Ejemplo:*

```
"coordinates":
{
    "coordinates":
    [
        -3.6402891,
        40.39208401
    ],
    "type":"Point"
}
```

- **created_at:** (*String*) Momento en el que el Tweet ha sido creado. La fecha y hora se rige por el estándar UTC[1]. *Ejemplo:*

```
created_at: "Thu Jun 26 21:08:01 +0000 2014"
```

[1]UTC (Coordinated Time Universal): Antiguamente fue llamada "la hora en el meridiano de Greenwich"("GMT") o el "tiempo Zulu"("Z"). Es la hora local en el Meridiano Primario (aquél cuya longitud es 0 grados) dada en horas y minutos en formato 24 horas.

- **current_user_retweet:** (*Object; Perspectival*)Solamente tiene valor si el campo *include_my_retweet* tiene valor true. Detalla el id del retweet de un Tweet, en caso de existir. *Ejemplo:*

```
"current_user_retweet": {
  "id": 26815871309,
  "id_str": "26815871309"
}
```

- **entities:** (*Entities*) Las entidades que han sido analizadas de forma separada al texto del Tweet. Incluye los hastags utilizados, las urls externas a Twitter y las menciones de usuarios. *Ejemplo:*

```
"entities":
{
    "hashtags":[],
    "urls":[],
    "user_mentions":[]
}
```

- **favorite_count:** (*Integer; Nullable*) Indica cuantas veces aproximadamente ha sido un Tweet marcado como favorito por los usuarios. *Ejemplo:*

```
"favorite_count":632
```

- **favorited:** (*Boolean; Nullable, Perspectival*) Indica si este Tweet ha sido marcado como favorito por el usuario que se autentica. *Ejemplo:*

```
"favorited":true
```

- **filter_level:** (*String*) Indica el valor máximo del parámetro *filter_level* que se refiere a la circulación del Tweet. Puede tener los valores: *none*, *low* y *medium*. *Ejemplo:*

```
"filter_level": "medium"
```

- **geo:** (*Object; Deprecated,Nullable*) Objeto que contiene las coordenadas.

- **id:** (*Int64*) Es la representación entera del identificador único para este Tweet. Este número es mayor que 53 bits. Algunos lenguajes de programación puede tener dificultades para interpretarlo. El uso de un entero con signo de 64 bits para almacenar este identificador es una forma segura. se utiliza el parámetro *id_str* para buscar este identificador. *Ejemplo:*

```
"id":482269071204184060
```

- **id_str:** (*String*)La representación en String del identificador único del Tweet. *Ejemplo:*

```
id_str: "482269071204184064"
```

- **in_reply_to_screen_name:** (*String; Nullable*) Si el Tweet es una respuesta a otro, este campo contiene el "screen name.º alias del autor el original de Tweet. *Ejemplo:*

  ```
  "in_reply_to_screen_name":"twitterapi"
  ```

- **in_reply_to_status_id:** (*Int64; Nullable*) Si el Tweet representado es una respuesta a otro, este campo contiene la representación entera del id de Tweet original. *Ejemplo:*

  ```
  "in_reply_to_status_id":114749583439036416
  ```

- **in_reply_to_status_id_str:** (*String; Nullable*) Si el Tweet representado es una respuesta a otro, este campo contiene la representación en String del id de Tweet original. *Ejemplo:*

  ```
  "in_reply_to_status_id_str":"114749583439036416"
  ```

- **in_reply_to_user_id:** (*Int64; Nullable*) Si el Tweet representado es una respuesta a otro, este campo contiene la representación en entera del id del autor del Tweet original. *Ejemplo:*

  ```
  "in_reply_to_user_id":819797
  ```

- **in_reply_to_user_id_str:** (*String; Nullable*) Si el Tweet representado es una respuesta a otro, este campo contiene la representación en String del id del autor del Tweet original. *Ejemplo:*

  ```
  "in_reply_to_user_id_str":"819797"
  ```

- **lang:** (*String; Nullable*)Cuando está presente, indica un identificador de idioma BCP 47[2] correspondiente al texto Tweet, o ünd"si el lenguaje no puede ser detectado. *Ejemplo:*

  ```
  "lang": "en"
  ```

- **place:** (*Places; Nullable*) Cuando está presente, indica que el tweet se asocia a un lugar, pero no necesariamente se origina en ese lugar. *Ejemplo:*

  ```
  "place":
  {
      "attributes":{},
       "bounding_box":
      {
  ```

[2] http://tools.ietf.org/html/bcp47

```
            "coordinates":
            [[
                    [-77.119759,38.791645],
                    [-76.909393,38.791645],
                    [-76.909393,38.995548],
                    [-77.119759,38.995548]
            ]],
            "type":"Polygon"
        },
         "country":"United States",
         "country_code":"US",
         "full_name":"Washington, DC",
         "id":"01fbe706f872cb32",
         "name":"Washington",
         "place_type":"city",
         "url": "http://api.twitter.com/1/geo/id/01fbe706f872cb32.json"
}
```


- **possibly_sensitive:** (*Boolean: Nullable*) Este campo surge solamente cuando un tweet contiene enlaces. Es un indicador de que la dirección que figura en el tweet puede incluir contenidos identificados como contenido sensible. *Ejemplo:*

  `"possibly_sensitive":true`

- **scopes:** (*Object*) Un conjunto de pares clave-valor que indican la entrega contextual prevista del Tweet. Actualmente es utilizado por los productos promocionados de Twitter.

  `"scopes":{"followers":false}`

- **retweet_count:** (*Int*) Número de veces que el Tweet a sido retwiteado.*Ejemplo:*

  `"retweet_count":322`

- **retweeted:** (*Boolean; Perspectival*) Indica si este Tweet ha retwiteado por el usuario que se autentica. *Ejemplo:*

  `"retweeted":false`

- **retweeted_status:** (*Tweet*) Los usuarios pueden ampliar la difusión de los Tweets escritos por otros usuarios mediante un retwit (retweet). Los retweets se pueden distinguir de los tweets típicos por la existencia de un atributo *retweeted_status*. Este atributo contiene una representación del Tweet original que fue retwiteado. Hay que tener en cuenta que los retwits del retwits no muestran representaciones del retwit intermediario, sino sólo del tweet original. *Ejemplo:*

- **source:** (*String*) Se utiliza para publicar el tweet en formato HTML. *Ejemplo:*

```
source: "<a href="http://twitter.com/download/iphone"
 rel="nofollow">Twitter for iPhone</a>"
```

- **text:** (*Nullable*) Texto UTF-8 de la actualización de estado.*Ejemplo:*

```
text: "Si la felicidad, como la gripe, es un estado
 q se transmite, contagia y propaga... Entonces causemos
 una epidemia!!"
```

- **truncated:** (*Boolean*)Indica si el valor del parámetro de texto se trunca. Puede darse el caso cuando un Retweet tiene una longitud superior a la longitud de 140 caracteres Tweet. El texto truncado terminará en puntos suspensivos. Twitter ahora rechaza los Tweets largos, por lo que la gran mayoría de los Tweets tendrá este parámetro a *false*. Hay que tener en cuenta que mientras los Retweets nativos pueden tener su propiedad de texto acortada, el texto original estará disponible bajo el objeto *retweeted_status* y el parámetro truncada se establece en el valor de la condición original (en la mayoría de los casos, false). *Ejemplo:*

```
"truncated":true
```

- **user:** (*Users*) El usuario que publicó este Tweet. Pordemos observar sus atributos en las siguientes líneas de código:

```
"user":{"statuses_count":3080,
"favourites_count":22,
"protected":false,
"profile_text_color":"437792",
"profile_image_url":"...",
"name":"Twitter API",
"profile_sidebar_fill_color":"a9d9f1",
"listed_count":9252,
"following":true,
"profile_background_tile":false,
"utc_offset":-28800,
"description":"The Real Twitter API. I tweet about API
changes, service issues and happily answer questions
about Twitter and our API. Don't get an answer?
It's on my website.",
"location":"San Francisco, CA",
"contributors_enabled":true,
"verified":true,
"profile_link_color":"0094C2",
"followers_count":665829,
"url":"http:\/\/dev.twitter.com",
"default_profile":false,
"profile_sidebar_border_color":"0094C2",
"screen_name":"twitterapi",
```

```
"default_profile_image":false,
"notifications":false,
"display_url":null,
"show_all_inline_media":false,
"geo_enabled":true,
"profile_use_background_image":true,
"friends_count":32, "id_str":"6253282",
"entities":{"hashtags":[],
"urls":[],
"user_mentions":[]},
"expanded_url":null,
"is_translator":false,
"lang":"en",
"time_zone":"Pacific Time (US &amp;amp; Canada)",
"created_at":"Wed May 23 06:01:13 +0000 2007",
"profile_background_color":"e8f2f7",
"id":6253282,
"follow_request_sent":false,
"profile_background_image_url_https":"...",
"profile_background_image_url":"...",
"profile_image_url_https":"..."}
```


- **withheld_copyright:** (*Boolean*) Cuando este parámetro se establece a *true*, indica que existe un contenido que se ha retenido debido a una reclamación de la DMCA. *Ejemplo:*


  ```
  "withheld_copyright": true
  ```


- **withheld_in_countries:** (*Array de String*) Cuando está presente, indica una lista de códigos de país de dos letras mayúsculas en donde este contenido es retenido.

  - "XX": El contenido es retenido en todos los países.
  - "XY": Contenido está retenido debido a una petición de DMCA.


  ```
  "withheld_in_countries": ["GR", "HK", "MY"]
  ```


- **withheld_scope:** (*String*) Cuando está presente, indica si el contenido que se está retenido es el "estado." "usuario".


  ```
  "withheld_scope": "status"
  ```
